```
   AAAAAAAAA     NNN         NNN     AAAAAAAAA     LLL              YYY         YYY  ZZZZZZZZZZZZZZZ
   AAAAAAAAA     NNN         NNN     AAAAAAAAA     LLL              YYY         YYY  ZZZZZZZZZZZZZZZ
   AAAAAAAAA     NNN         NNN     AAAAAAAAA     LLL              YYY         YYY  ZZZZZZZZZZZZZZZ
AAA         AAA  NNN         NNN  AAA         AAA  LLL              YYY         YYY              ZZZ
AAA         AAA  NNN         NNN  AAA         AAA  LLL              YYY         YYY              ZZZ
AAA         AAA  NNN         NNN  AAA         AAA  LLL              YYY         YYY              ZZZ
AAA         AAA  NNNNNN      NNN  AAA         AAA  LLL                 YYY   YYY              ZZZ
AAA         AAA  NNNNNN      NNN  AAA         AAA  LLL                 YYY   YYY              ZZZ
AAA         AAA  NNNNNN      NNN  AAA         AAA  LLL                 YYY   YYY              ZZZ
AAA         AAA  NNN   NNN   NNN  AAA         AAA  LLL                    YYY              ZZZ
AAA         AAA  NNN   NNN   NNN  AAA         AAA  LLL                    YYY              ZZZ
AAA         AAA  NNN   NNN   NNN  AAA         AAA  LLL                    YYY              ZZZ
AAAAAAAAAAAAAAA  NNN      NNNNNN  AAAAAAAAAAAAAAA  LLL                    YYY           ZZZ
AAAAAAAAAAAAAAA  NNN      NNNNNN  AAAAAAAAAAAAAAA  LLL                    YYY           ZZZ
AAAAAAAAAAAAAAA  NNN      NNNNNN  AAAAAAAAAAAAAAA  LLL                    YYY           ZZZ
AAA         AAA  NNN         NNN  AAA         AAA  LLL                    YYY        ZZZ
AAA         AAA  NNN         NNN  AAA         AAA  LLL                    YYY        ZZZ
AAA         AAA  NNN         NNN  AAA         AAA  LLL                    YYY        ZZZ
AAA         AAA  NNN         NNN  AAA         AAA  LLLLLLLLLLLLLLL        YYY        ZZZZZZZZZZZZZZZ
AAA         AAA  NNN         NNN  AAA         AAA  LLLLLLLLLLLLLLL        YYY        ZZZZZZZZZZZZZZZ
AAA         AAA  NNN         NNN  AAA         AAA  LLLLLLLLLLLLLLL        YYY        ZZZZZZZZZZZZZZZ
```

```
RRRRRRRR    MM       MM    SSSSSSSS  RRRRRRRR    EEEEEEEEEE  PPPPPPPP      000000    RRRRRRRR    TTTTTTTTTT
RRRRRRRR    MM       MM    SSSSSSSS  RRRRRRRR    EEEEEEEEEE  PPPPPPPP      000000    RRRRRRRR    TTTTTTTTTT
RR      RR  MMMM   MMMM  SS          RR      RR  EE          PP      PP  00      00  RR      RR      TT
RR      RR  MMMM   MMMM  SS          RR      RR  EE          PP      PP  00      00  RR      RR      TT
RR      RR  MM MM MM MM  SS          RR      RR  EE          PP      PP  00      00  RR      RR      TT
RR      RR  MM MM MM MM  SS          RR      RR  EE          PP      PP  00      00  RR      RR      TT
RRRRRRRR    MM  MMM  MM    SSSSSS    RRRRRRRR    EEEEEEEE    PPPPPPPP    00      00  RRRRRRRR        TT
RRRRRRRR    MM  MMM  MM    SSSSSS    RRRRRRRR    EEEEEEEE    PPPPPPPP    00      00  RRRRRRRR        TT
RR  RR      MM       MM          SS  RR  RR      EE          PP          00      00  RR  RR          TT
RR  RR      MM       MM          SS  RR  RR      EE          PP          00      00  RR  RR          TT
RR    RR    MM       MM          SS  RR    RR    EE          PP          00      00  RR    RR        TT          ....
RR    RR    MM       MM          SS  RR    RR    EE          PP          00      00  RR    RR        TT          ....
RR      RR  MM       MM  SSSSSSSS    RR      RR  EEEEEEEEEE  PP            000000    RR      RR      TT          ....
RR      RR  MM       MM  SSSSSSSS    RR      RR  EEEEEEEEEE  PP            000000    RR      RR      TT          ....

LL              IIIIII      SSSSSSSS
LL              IIIIII      SSSSSSSS
LL                II      SS
LL                II      SS
LL                II      SS
LL                II      SS
LL                II        SSSSSS
LL                II        SSSSSS
LL                II              SS
LL                II              SS
LL                II              SS
LL                II              SS
LLLLLLLLLL      IIIIII    SSSSSSSS
LLLLLLLLLL      IIIIII    SSSSSSSS
```

```
;  1          0001  0 %title 'RMSREPORT - Handle Output for ANALYZE/RMS_FILE'
;  2          0002  0         module rmsreport (
;  3          0003  1                           ident='V04-000') = begin
;  4          0004  1
;  5          0005  1 !
;  6          0006  1 !*****************************************************************************
;  7          0007  1 !*                                                                           *
;  8          0008  1 !*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                  *
;  9          0009  1 !*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                    *
;  10         0010  1 !*  ALL RIGHTS RESERVED.                                                      *
;  11         0011  1 !*                                                                           *
;  12         0012  1 !*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED   *
;  13         0013  1 !*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE  *
;  14         0014  1 !*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER  *
;  15         0015  1 !*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY  *
;  16         0016  1 !*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY  *
;  17         0017  1 !*  TRANSFERRED.                                                              *
;  18         0018  1 !*                                                                           *
;  19         0019  1 !*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE   *
;  20         0020  1 !*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT   *
;  21         0021  1 !*  CORPORATION.                                                              *
;  22         0022  1 !*                                                                           *
;  23         0023  1 !*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS   *
;  24         0024  1 !*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                   *
;  25         0025  1 !*                                                                           *
;  26         0026  1 !*                                                                           *
;  27         0027  1 !*****************************************************************************
;  28         0028  1 !
;  29         0029  1
;  30         0030  1 !++
;  31         0031  1 ! Facility:     VAX/VMS Analyze Facility, Handle Reports for ANALYZE/RMS_FILE
;  32         0032  1 !
;  33         0033  1 ! Abstract:     This module is responsible for all reports from the
;  34         0034  1 !               ANALYZE/RMS_FILE command.  Reports can be routed to a file
;  35         0035  1 !               and/or the terminal.
;  36         0036  1 !
;  37         0037  1 !
;  38         0038  1 ! Environment:
;  39         0039  1 !
;  40         0040  1 ! Author: Paul C. Anagnostopoulos, Creation Date:  18 February 1981
;  41         0041  1 !
;  42         0042  1 ! Modified By:
;  43         0043  1 !
;  44         0044  1 !       V03-009 DGB0055         Donald G. Blair         14-Jun-1984
;  45         0045  1 !               On ANLRMS$_OPENOUT error, rather than print the file
;  46         0046  1 !               spec from the /OUT qualifier (which may be null), print
;  47         0047  1 !               the expanded file spec derived therefrom.
;  48         0048  1 !
;  49         0049  1 !       V03-008 DGB0045         Donald G. Blair         08-May-1984
;  50         0050  1 !               Incorporate the routine ANL$EXIT_WITH_STATUS into the main
;  51         0051  1 !               routine ANL$RMS and add handling for ANL$WORST_ERROR
;  52         0052  1 !               to ANL$FORMAT_ERROR as part of fixing ANALYZRMS so
;  53         0053  1 !               it returns status correctly.
;  54         0054  1 !
;  55         0055  1 !       V03-007 RRB0003         Rowland R. Bradley      1-Jan-1984
;  56         0056  1 !               Correct "value required context" error in ANL$EXIT_WITH_STATUS
;  57         0057  1 !
```

```
;    58          0058  1 ;       V03-006 PCA1012         Paul C. Anagnostopoulos  6-Apr-1983
;    59          0059  1 ;               Add support for /NOOUTPUT qualifier.
;    60          0060  1 ;
;    61          0061  1 ;       V03-005 PCA1011         Paul C. Anagnostopoulos  1-Apr-1983
;    62          0062  1 ;               Change the message prefix to ANLRMS$_ to ensure that
;    63          0063  1 ;               message symbols are unique across all ANALYZEs.  This
;    64          0064  1 ;               is necessitated by the new merged message files.
;    65          0065  1 ;
;    66          0066  1 ;       V03-004 PCA1001         Paul C. Anagnostopoulos  4-Nov-1982
;    67          0067  1 ;               Display the global buffer count for all files, not
;    68          0068  1 ;               just relative and indexed ones.
;    69          0069  1 ;               Add code to support the new /SUMMARY mode.
;    70          0070  1 ;
;    71          0071  1 ;       V03-003 PCA0031         Paul Anagnostopoulos    24-Mar-1982
;    72          0072  1 ;               Fix error messages so they use the correct STV value.
;    73          0073  1 ;
;    74          0074  1 ;       V03-002 PCA0012         Paul Anagnostopoulos    16-Mar-1982
;    75          0075  1 ;               Remove maximum record size restriction on report file.
;    76          0076  1 ;
;    77          0077  1 ;       V03-001 PCA0011         Paul Anagnostopoulos    16-Mar-1982
;    78          0078  1 ;               Include new global buffer count when formatting the
;    79          0079  1 ;               report of the file attribute area.
;    80          0080  1 ;--
```

```
;    82          0081  1 %sbttl 'Module Declarations'
;    83          0082  1 !
;    84          0083  1 ! Libraries and Requires:
;    85          0084  1 !
;    86          0085  1
;    87          0086  1 library 'lib';
;    88          0087  1 require 'rmsreq';
;    89          0596  1
;    90          0597  1 !
;    91          0598  1 ! Table of Contents:
;    92          0599  1 !
;    93          0600  1
;    94          0601  1 forward routine
;    95          0602  1         anl$prepare_report_file: novalue,
;    96          0603  1         anl$report_page: novalue,
;    97          0604  1         anl$format_line: novalue,
;    98          0605  1         anl$format_skip: novalue,
;    99          0606  1         anl$format_error: novalue,
;   100          0607  1         anl$error_count: novalue,
;   101          0608  1         anl$format_flags: novalue,
;   102          0609  1         anl$format_hex: novalue,
;   103          0610  1         anl$format_protection_mask: novalue,
;   104          0611  1         anl$format_file_attributes: novalue;
;   105          0612  1
;   106          0613  1 !
;   107          0614  1 ! External References:
;   108          0615  1 !
;   109          0616  1
;   110          0617  1 external routine
;   111          0618  1         cli$get_value: addressing_mode(general),
;   112          0619  1         cli$present: addressing_mode(general),
;   113          0620  1         lib$lp_lines: addressing_mode(general),
;   114          0621  1         lib$put_output: addressing_mode(general),
;   115          0622  1         str$trim: addressing_mode(general);
;   116          0623  1
;   117          0624  1 external
;   118          0625  1         anl$gb_mode: byte,
;   119          0626  1         anl$gl_fat: ref block[,byte];
;   120          0627  1
;   121          0628  1 !
;   122          0629  1 ! Own Variables:
;   123          0630  1 !
;   124          0631  1 ! To create the report file, we need a RAB, FAB, and NAM block.  We also
;   125          0632  1 ! need a second NAM block to act as the related NAM block.
;   126          0633  1
;   127          0634  1 own
;   128          0635  1         own_described_buffer(expanded_spec,nam$c_maxrss),
;   129          0636  1         related_resultant_spec: block[nam$c_maxrss,byte],
;   130          0637  1         related_expanded_spec: block[nam$c_maxrss,byte],
;   131          0638  1
;   132        P 0639  1         related_nam: $nam(esa=related_expanded_spec,
;   133        P 0640  1                          ess=nam$c_maxrss,
;   134        P 0641  1                          rsa=related_resultant_spec,
;   135          0642  1                          rss=nam$c_maxrss),
;   136          0643  1
;   137        P 0644  1         report_nam: $nam(rlf=related_nam,
;   138        P 0645  1                         esa=expanded_spec+8,
```

```
;  139          0646  1                               ess=nam$c_maxrss),
;  140          0647  1
;  141          0648  1          own_described_buffer(report_file_spec,nam$c_maxrss),
;  142          0649  1
;  143        P 0650  1          report_fab: $fab(fac=put,
;  144        P 0651  1                          fop=ofp,
;  145        P 0652  1                          nam=report_nam,
;  146        P 0653  1                          org=seq,
;  147        P 0654  1                          rat=cr,
;  148          0655  1                          rfm=var),
;  149          0656  1
;  150        P 0657  1          report_rab: $rab(fab=report_fab,
;  151          0658  1                          rac=seq);
;  152          0659  1
;  153          0660  1 ! The following variables are needed to format the report.
;  154          0661  1
;  155          0662  1 own
;  156          0663  1          generating_report: byte,
;  157          0664  1          report_heading_msg: long,
;  158          0665  1          own_described_buffer(input_file_spec,nam$c_maxrss),
;  159          0666  1          page_number: long,
;  160          0667  1          line_counter: signed long;
;  161          0668  1
;  162          0669  1 ! We need some variables for keeping track of errors.  One tells us where
;  163          0670  1 ! the analysis report is going.  We also count the number of errors.
;  164          0671  1
;  165          0672  1 own
;  166          0673  1          report_to_file: byte,
;  167          0674  1          error_count: long initial(0);
```

```
;  169          0675  1 %sbttl 'ANL$PREPARE_REPORT_FILE - Prepare Report File'
;  170          0676  1 !++
;  171          0677  1 ! Functional Description:
;  172          0678  1 !       This routine is called whenever we begin the analysis of a new
;  173          0679  1 !       file.  On the first call, it creates a report file to receive
;  174          0680  1 !       the analysis.  On subsequent calls, if any, it just starts a new
;  175          0681  1 !       report in the file.
;  176          0682  1 !
;  177          0683  1 ! Formal Parameters:
;  178          0684  1 !       heading_msg     An optional message code specifying the report
;  179          0685  1 !                       page heading message.
;  180          0686  1 !       input_spec      The resultant spec of the input file we are analyzing.
;  181          0687  1 !
;  182          0688  1 ! Implicit Inputs:
;  183          0689  1 !       global data
;  184          0690  1 !
;  185          0691  1 ! Implicit Outputs:
;  186          0692  1 !       global data
;  187          0693  1 !
;  188          0694  1 ! Returned Value:
;  189          0695  1 !       none
;  190          0696  1 !
;  191          0697  1 ! Side Effects:
;  192          0698  1 !
;  193          0699  1 !--
;  194          0700  1
;  195          0701  1
;  196          0702  2 global routine anl$prepare_report_file(heading_msg,input_spec): novalue = begin
;  197          0703  2
;  198          0704  2 bind
;  199          0705  2         input_spec_dsc = .input_spec: descriptor;
;  200          0706  2
;  201          0707  2 own
;  202          0708  2         first_call: byte initial(true);
;  203          0709  2
;  204          0710  2 local
;  205          0711  2         status: long;
;  206          0712  2
;  207          0713  2
;  208          0714  2
;  209          0715  2 ! Save the input file spec for use in the report page headings.
;  210          0716  2
;  211          0717  2 input_file_spec[len] = .input_spec_dsc[len];
;  212          0718  2 ch$move(.input_spec_dsc[len],.input_spec_dsc[ptr], .input_file_spec[ptr]);
;  213          0719  2
;  214          0720  2 ! See if we are to generate a report.  If not, we can just leave.
;  215          0721  2
;  216          0722  2 generating_report = cli$present(describe('OUTPUT'));
;  217          0723  2 if not .generating_report then
;  218          0724  2         return;
;  219          0725  2
;  220          0726  2 ! If this is the first call, then we need to create the report file and
;  221          0727  2 ! prepare for one or more analysis reports.
;  222          0728  2
;  223          0729  3 if .first_call then (
;  224          0730  3
;  225          0731  3         ! We begin by obtaining the value of the /OUTPUT qualifier.  This will
```

```
;  226          0732  3          ! tell us the name of the desired report file.  Trim the name for use
;  227          0733  3          ! in error messages.
;  228          0734  3
;  229          0735  3          report_to_file = cli$get_value(describe('OUTPUT'),report_file_spec);
;  230          0736  3          str$trim(report_file_spec,report_file_spec,report_file_spec);
;  231          0737  3
;  232          0738  3          ! Now we split up depending on the mode of operation.
;  233          0739  3
;  234          0740  3          selectoneu .anl$gb_mode of set
;  235          0741  3
;  236          0742  3          [anl$k_check,
;  237          0743  3           anl$k_statistics,
;  238          0744  3           anl$k_summary]:
;  239          0745  3
;  240          0746  3                  ! In these modes, the user specifies the name of the
;  241          0747  3                  ! report file, and we use ANALYZE.ANL as the defaults.
;  242          0748  3                  ! If the user didn't include a value on the /OUTPUT qualifier,
;  243          0749  3                  ! then we just put the report on the terminal.
;  244          0750  3
;  245          0751  4                  if .report_to_file then (
;  246          0752  4                          report_fab[fab$l_fna] = .report_file_spec[ptr];
;  247          0753  4                          report_fab[fab$b_fns] = .report_file_spec[len];
;  248          0754  4                          report_fab[fab$l_dna] = uplit byte('ANALYZE.ANL');
;  249          0755  4                          report_fab[fab$b_dns] = 11;
;  250          0756  4                  ) else (
;  251          0757  4                          report_fab[fab$l_fna] = uplit byte('SYS$OUTPUT');
;  252          0758  4                          report_fab[fab$b_fns] = 10;
;  253          0759  3                  );
;  254          0760  3
;  255          0761  3          [anl$k_fdl]:
;  256          0762  3
;  257          0763  3                  ! In this mode, the user specifies the name of the FDL
;  258          0764  3                  ! file, we use .FDL as the default, and we use a related
;  259          0765  3                  ! name equal to the input file spec.  This produces the
;  260          0766  3                  ! standard related name situation where the output file
;  261          0767  3                  ! has the same name as the input file.
;  262          0768  3
;  263          0769  3                  ! To parse the input file name, we use the report FAB
;  264          0770  3                  ! temporarily so we can do a $PARSE and a $SEARCH into
;  265          0771  3                  ! the related NAM block.
;  266          0772  3
;  267          0773  4                  (report_fab[fab$l_fna] = .input_spec_dsc[ptr];
;  268          0774  4                  report_fab[fab$b_fns] = .input_spec_dsc[len];
;  269          0775  4                  report_fab[fab$l_nam] = related_nam;
;  270          0776  4                  status = $parse(fab=report_fab);
;  271          0777  4                  check (.status, .status);
;  272          0778  4                  status = $search(fab=report_fab);
;  273          0779  4                  check (.status, .status);
;  274          0780  4
;  275          0781  4                  ! Now we can set up the blocks for creation of the report file.
;  276          0782  4                  ! The FAB specifies output file parse, as required.
;  277          0783  4
;  278          0784  4                  report_fab[fab$l_fna] = .report_file_spec[ptr];
;  279          0785  4                  report_fab[fab$b_fns] = .report_file_spec[len];
;  280          0786  4                  report_fab[fab$l_dna] = uplit byte('.FDL');
;  281          0787  4                  report_fab[fab$b_dns] = 4;
;  282          0788  3                  report_fab[fab$l_nam] = report_nam;);
```

```
;  283          0789  3
;  284          0790  3          [anl$k_interactive]:
;  285          0791  3
;  286          0792  3                  ! In this mode, the user specifies the name of the
;  287          0793  3                  ! transcipt file, and we use ANALYZE.ANL as the defaults.
;  288          0794  3                  ! If the user didn't include a value on the /OUTPUT qualifier,
;  289          0795  3                  ! then we don't produce a transcript.
;  290          0796  3
;  291          0797  4                  if .report_to_file then (
;  292          0798  4                          report_fab[fab$l_fna] = .report_file_spec[ptr];
;  293          0799  4                          report_fab[fab$b_fns] = .report_file_spec[len];
;  294          0800  4                          report_fab[fab$l_dna] = uplit byte('ANALYZE.ANL');
;  295          0801  4                          report_fab[fab$b_dns] = 11;
;  296          0802  4                  ) else
;  297          0803  4                          return;
;  298          0804  3
;  299          0805  3          tes;
;  300          0806  3
;  301          0807  3          ! Now we can create the report file and connect the RAB.
;  302          0808  3
;  303          0809  3          status = $create(fab=report_fab);
;  304          0810  3          expanded_spec[len] = .report_nam[nam$b_esl];
;  305          0811  3          check (.status, anlrms$_openout,1,expanded_spec,.status,.report_fab[fab$l_stv]);
;  306          0812  3          status = $connect(rab=report_rab);
;  307          0813  3          check (.status, .status);
;  308          0814  3
;  309          0815  3          ! Save the heading message code.
;  310          0816  3
;  311          0817  3          report_heading_msg = .heading_msg;
;  312          0818  3
;  313          0819  3          first_call = false;
;  314          0820  2 );
;  315          0821  2
;  316          0822  2 ! Begin the report by resetting the page number and starting a new page.
;  317          0823  2
;  318          0824  2 page_number = 0;
;  319          0825  2 anl$report_page();
;  320          0826  2
;  321          0827  2 return;
;  322          0828  2
;  323          0829  1 end;


                                                                .TITLE  RMSREPORT RMSREPORT - Handle Output for ANALYZE
;                                                                                 /RMS_FILE
                                                                .IDENT  \V04-000\

                                                                .PSECT  $PLIT$,NOWRT,NOEXE,2

                                54  55  50  54  55  4F  00000 P.AAB:  .ASCII  \OUTPUT\                                    ;
                                                        00006           .BLKB   2
                                            00000006    00008 P.AAA:  .LONG   6                                           ;
                                            00000000'   0000C           .ADDRESS P.AAB                                    ;
                                54  55  50  54  55  4F  00010 P.AAD:  .ASCII  \OUTPUT\                                    ;
                                                        00016           .BLKB   2
                                            00000006    00018 P.AAC:  .LONG   6                                           ;
                                            00000000'   0001C           .ADDRESS P.AAD                                    ;
```

```
                                                                D 2
4C 4E 41 2E 45 5A 59 4C 41 4E 41  00020 P.AAE:   .ASCII  \ANALYZE.ANL\
   54 55 50 54 55 4F 24 53 59 53  0002B P.AAF:   .ASCII  \SYS$OUTPUT\
                     4C 44 46 2E  00035 P.AAG:   .ASCII  \.FDL\
4C 4E 41 2E 45 5A 59 4C 41 4E 41  00039 P.AAH:   .ASCII  \ANALYZE.ANL\

                                              .PSECT  $OWN$,NOEXE,2

                                000000FF  00000 EXPANDED_SPEC:
                                                .LONG   255
                                00000000' 00004         .ADDRESS EXPANDED_SPEC+8
                                          00008         .BLKB   255
                                          00107         .BLKB   1
                                          00108 RELATED_RESULTANT_SPEC:
                                                .BLKB   255
                                          00207         .BLKB   1
                                          00208 RELATED_EXPANDED_SPEC:
                                                .BLKB   255
                                          00307         .BLKB   1
                                       02 00308 RELATED_NAM:
                                                .BYTE   2
                                       60 00309         .BYTE   96
                                       FF 0030A         .BYTE   -1
                                       00 0030B         .BYTE   0
                                00000000' 0030C         .ADDRESS RELATED_RESULTANT_SPEC
                                       00 00310         .BYTE   0
                                       00 00311         .BYTE   0
                                       FF 00312         .BYTE   -1
                                       00 00313         .BYTE   0
                                00000000' 00314         .ADDRESS RELATED_EXPANDED_SPEC
                                00000000  00318         .LONG   0
                                    0000# 0031C         .WORD   0[8]
                                    0000# 0032C         .WORD   0[3]
                                    0000# 00332         .WORD   0[3]
                                00000000  00338         .LONG   0
                                00000000  0033C         .LONG   0
                                       00 00340         .BYTE   0
                                       00 00341         .BYTE   0
                                       00 00342         .BYTE   0
                                       00 00343         .BYTE   0
                                       00 00344         .BYTE   0
                                       00 00345         .BYTE   0
                                       00# 00346        .BYTE   0[2]
                                00000000  00348         .LONG   0
                                00000000  0034C         .LONG   0
                                00000000  00350         .LONG   0
                                00000000  00354         .LONG   0
                                00000000  00358         .LONG   0
                                00000000  0035C         .LONG   0
                                00000000# 00360         .LONG   0[2]
                                       02 00368 REPORT_NAM:
                                                .BYTE   2
                                       60 00369         .BYTE   96
                                       00 0036A         .BYTE   0
                                       00 0036B         .BYTE   0
                                00000000  0036C         .LONG   0
                                       00 00370         .BYTE   0
                                       00 00371         .BYTE   0
```

```
       FF  00372               .BYTE    -1                                   ;
       00  00373               .BYTE    0                                    ;
00000000'  00374               .ADDRESS EXPANDED_SPEC+8                      ;
00000000'  00378               .ADDRESS RELATED_NAM                          ;
    0000#  0037C               .WORD    0[8]                                 ;
    0000#  0038C               .WORD    0[3]                                 ;
    0000#  00392               .WORD    0[3]                                 ;
 00000000  00398               .LONG    0                                    ;
 00000000  0039C               .LONG    0                                    ;
       00  003A0               .BYTE    0                                    ;
       00  003A1               .BYTE    0                                    ;
       00  003A2               .BYTE    0                                    ;
       00  003A3               .BYTE    0                                    ;
       00  003A4               .BYTE    0                                    ;
       00  003A5               .BYTE    0                                    ;
      00#  003A6               .BYTE    0[2]                                 ;
 00000000  003A8               .LONG    0                                    ;
 00000000  003AC               .LONG    0                                    ;
 00000000  003B0               .LONG    0                                    ;
 00000000  003B4               .LONG    0                                    ;
 00000000  003B8               .LONG    0                                    ;
 00000000  003BC               .LONG    0                                    ;
00000000#  003C0               .LONG    0[2]                                 ;
 000000FF  003C8 REPORT_FILE_SPEC:
                               .LONG    255                                  ;
00000000'  003CC               .ADDRESS REPORT_FILE_SPEC+8                   ;
           003D0               .BLKB    255
           004CF               .BLKB    1
       03  004D0 REPORT_FAB:
                               .BYTE    3                                    ;
       50  004D1               .BYTE    80                                   ;
     0000  004D2               .WORD    0                                    ;
 20000000  004D4               .LONG    536870912                            ;
 00000000  004D8               .LONG    0                                    ;
 00000000  004DC               .LONG    0                                    ;
 00000000  004E0               .LONG    0                                    ;
     0000  004E4               .WORD    0                                    ;
       01  004E6               .BYTE    1                                    ;
       00  004E7               .BYTE    0                                    ;
 00000000  004E8               .LONG    0                                    ;
       00  004EC               .BYTE    0                                    ;
       00  004ED               .BYTE    0                                    ;
       02  004EE               .BYTE    2                                    ;
       02  004EF               .BYTE    2                                    ;
 00000000  004F0               .LONG    0                                    ;
 00000000  004F4               .LONG    0                                    ;
00000000'  004F8               .ADDRESS REPORT_NAM                           ;
 00000000  004FC               .LONG    0                                    ;
 00000000  00500               .LONG    0                                    ;
       00  00504               .BYTE    0                                    ;
       00  00505               .BYTE    0                                    ;
     0000  00506               .WORD    0                                    ;
 00000000  00508               .LONG    0                                    ;
     0000  0050C               .WORD    0                                    ;
       00  0050E               .BYTE    0                                    ;
       00  0050F               .BYTE    0                                    ;
 00000000  00510               .LONG    0                                    ;
```

```
        00000000  00514          .LONG    0                                        ;
            0000  00518          .WORD    0                                        ;
              00  0051A          .BYTE    0                                        ;
              00  0051B          .BYTE    0                                        ;
        00000000  0051C          .LONG    0                                        ;
              01  00520 REPORT_RAB:
                                 .BYTE    1                                        ;
              44  00521          .BYTE    68                                       ;
            0000  00522          .WORD    0                                        ;
        00000000  00524          .LONG    0                                        ;
        00000000  00528          .LONG    0                                        ;
        00000000  0052C          .LONG    0                                        ;
           0000#  00530          .WORD    0[3]                                     ;
            0000  00536          .WORD    0                                        ;
        00000000  00538          .LONG    0                                        ;
            0000  0053C          .WORD    0                                        ;
              00  0053E          .BYTE    0                                        ;
              00  0053F          .BYTE    0                                        ;
            0000  00540          .WORD    0                                        ;
            0000  00542          .WORD    0                                        ;
        00000000  00544          .LONG    0                                        ;
        00000000  00548          .LONG    0                                        ;
        00000000  0054C          .LONG    0                                        ;
        00000000  00550          .LONG    0                                        ;
              00  00554          .BYTE    0                                        ;
              00  00555          .BYTE    0                                        ;
              00  00556          .BYTE    0                                        ;
              00  00557          .BYTE    0                                        ;
        00000000  00558          .LONG    0                                        ;
       00000000'  0055C          .ADDRESS REPORT_FAB                               ;
        00000000  00560          .LONG    0                                        ;
                  00564 GENERATING_REPORT:
                                 .BLKB    1
                  00565          .BLKB    3
                  00568 REPORT_HEADING_MSG:
                                 .BLKB    4
        000000FF  0056C INPUT_FILE_SPEC:
                                 .LONG    255                                      ;
       00000000'  00570          .ADDRESS INPUT_FILE_SPEC+8                        ;
                  00574          .BLKB    255
                  00673          .BLKB    1
                  00674 PAGE_NUMBER:
                                 .BLKB    4
                  00678 LINE_COUNTER:
                                 .BLKB    4
                  0067C REPORT_TO_FILE:
                                 .BLKB    1
                  0067D          .BLKB    3
        00000000  00680 ERROR_COUNT:
                                 .LONG    0                                        ;
              01  00684 FIRST_CALL:
                                 .BYTE    1                                        ;

                                 .EXTRN   ANLRMS$_OK, ANLRMS$_ALLOC
                                 .EXTRN   ANLRMS$_ANYTHING
                                 .EXTRN   ANLRMS$_BACKUP, ANLRMS$_BKT
                                 .EXTRN   ANLRMS$_BKTAREA
```

```
.EXTRN  ANLRMS$_BKTCHECK
.EXTRN  ANLRMS$_BKTFLAGS
.EXTRN  ANLRMS$_BKTFREE
.EXTRN  ANLRMS$_BKTKEY, ANLRMS$_BKTLEVEL
.EXTRN  ANLRMS$_BKTNEXT
.EXTRN  ANLRMS$_BKTPTRSIZE
.EXTRN  ANLRMS$_BKTRECID
.EXTRN  ANLRMS$_BKTRECID3
.EXTRN  ANLRMS$_BKTSAMPLE
.EXTRN  ANLRMS$_BKTVBNFREE
.EXTRN  ANLRMS$_BUCKETSIZE
.EXTRN  ANLRMS$_CELL, ANLRMS$_CELLDATA
.EXTRN  ANLRMS$_CELLFLAGS
.EXTRN  ANLRMS$_CHECKHDG
.EXTRN  ANLRMS$_CONTIG, ANLRMS$_CREATION
.EXTRN  ANLRMS$_CTLSIZE
.EXTRN  ANLRMS$_DATAREC
.EXTRN  ANLRMS$_DATABKTVBN
.EXTRN  ANLRMS$_DUMPHEADING
.EXTRN  ANLRMS$_EOF, ANLRMS$_ERRORCOUNT
.EXTRN  ANLRMS$_ERRORNONE
.EXTRN  ANLRMS$_ERRORS, ANLRMS$_EXPIRATION
.EXTRN  ANLRMS$_FILEATTR
.EXTRN  ANLRMS$_FILEHDR
.EXTRN  ANLRMS$_FILEID, ANLRMS$_FILEORG
.EXTRN  ANLRMS$_FILESPEC
.EXTRN  ANLRMS$_FLAG, ANLRMS$_GLOBALBUFS
.EXTRN  ANLRMS$_HEXDATA
.EXTRN  ANLRMS$_HEXHEADING1
.EXTRN  ANLRMS$_HEXHEADING2
.EXTRN  ANLRMS$_IDXAREA
.EXTRN  ANLRMS$_IDXAREAALLOC
.EXTRN  ANLRMS$_IDXAREABKTSZ
.EXTRN  ANLRMS$_IDXAREANEXT
.EXTRN  ANLRMS$_IDXAREANOALLOC
.EXTRN  ANLRMS$_IDXAREAQTY
.EXTRN  ANLRMS$_IDXAREARECL
.EXTRN  ANLRMS$_IDXAREAUSED
.EXTRN  ANLRMS$_IDXKEY, ANLRMS$_IDXKEYAREAS
.EXTRN  ANLRMS$_IDXKEYBKTSZ
.EXTRN  ANLRMS$_IDXKEYBYTES
.EXTRN  ANLRMS$_IDXKEYITYPE
.EXTRN  ANLRMS$_IDXKEYDATAVBN
.EXTRN  ANLRMS$_IDXKEYFILL
.EXTRN  ANLRMS$_IDXKEYFLAGS
.EXTRN  ANLRMS$_IDXKEYKEYSZ
.EXTRN  ANLRMS$_IDXKEYNAME
.EXTRN  ANLRMS$_IDXKEYNEXT
.EXTRN  ANLRMS$_IDXKEYMINREC
.EXTRN  ANLRMS$_IDXKEYNULL
.EXTRN  ANLRMS$_IDXKEYPOSS
.EXTRN  ANLRMS$_IDXKEYROOTLVL
.EXTRN  ANLRMS$_IDXKEYROOTVBN
.EXTRN  ANLRMS$_IDXKEYSEGS
.EXTRN  ANLRMS$_IDXKEYSIZES
.EXTRN  ANLRMS$_IDXPRIMREC
.EXTRN  ANLRMS$_IDXPRIMRECFLAGS
```

```
                                                        .EXTRN  ANLRMS$_IDXPRIMRECID
                                                        .EXTRN  ANLRMS$_IDXPRIMRECLEN
                                                        .EXTRN  ANLRMS$_IDXPRIMRECRRV
                                                        .EXTRN  ANLRMS$_IDXPROAREAS
                                                        .EXTRN  ANLRMS$_IDXPROLOG
                                                        .EXTRN  ANLRMS$_IDXREC, ANLRMS$_IDXRECPTR
                                                        .EXTRN  ANLRMS$_IDXSIDR
                                                        .EXTRN  ANLRMS$_IDXSIDRDUPCNT
                                                        .EXTRN  ANLRMS$_IDXSIDRFLAGS
                                                        .EXTRN  ANLRMS$_IDXSIDRRECID
                                                        .EXTRN  ANLRMS$_IDXSIDRPTRFLAGS
                                                        .EXTRN  ANLRMS$_IDXSIDRPTRREF
                                                        .EXTRN  ANLRMS$_INTERCOMMAND
                                                        .EXTRN  ANLRMS$_INTERHDG
                                                        .EXTRN  ANLRMS$_LONGREC
                                                        .EXTRN  ANLRMS$_MAXRECSIZE
                                                        .EXTRN  ANLRMS$_NOBACKUP
                                                        .EXTRN  ANLRMS$_NOEXPIRATION
                                                        .EXTRN  ANLRMS$_NOSPANFILLER
                                                        .EXTRN  ANLRMS$_PERFORM
                                                        .EXTRN  ANLRMS$_PROLOGFLAGS
                                                        .EXTRN  ANLRMS$_PROLOGVER
                                                        .EXTRN  ANLRMS$_PROT, ANLRMS$_RECATTR
                                                        .EXTRN  ANLRMS$_RECFMT, ANLRMS$_RECLAIMBKT
                                                        .EXTRN  ANLRMS$_RELBUCKET
                                                        .EXTRN  ANLRMS$_RELEOFVBN
                                                        .EXTRN  ANLRMS$_RELMAXREC
                                                        .EXTRN  ANLRMS$_RELPROLOG
                                                        .EXTRN  ANLRMS$_RELIAB, ANLRMS$_REVISION
                                                        .EXTRN  ANLRMS$_STATHDG
                                                        .EXTRN  ANLRMS$_SUMMARYHDG
                                                        .EXTRN  ANLRMS$_OWNERUIC
                                                        .EXTRN  ANLRMS$_JNL, ANLRMS$_AIJNL
                                                        .EXTRN  ANLRMS$_BIJNL, ANLRMS$_ATJNL
                                                        .EXTRN  ANLRMS$_ATTOP, ANLRMS$_BADCMD
                                                        .EXTRN  ANLRMS$_BADPATH
                                                        .EXTRN  ANLRMS$_BADVBN, ANLRMS$_DOWNHELP
                                                        .EXTRN  ANLRMS$_DOWNPATH
                                                        .EXTRN  ANLRMS$_EMPTYBKT
                                                        .EXTRN  ANLRMS$_NODATA, ANLRMS$_NODOWN
                                                        .EXTRN  ANLRMS$_NONEXT, ANLRMS$_NORECLAIMED
                                                        .EXTRN  ANLRMS$_NORECS, ANLRMS$_NORRV
                                                        .EXTRN  ANLRMS$_RESTDONE
                                                        .EXTRN  ANLRMS$_STACKFULL
                                                        .EXTRN  ANLRMS$_UNINITINDEX
                                                        .EXTRN  ANLRMS$_FDLIDENT
                                                        .EXTRN  ANLRMS$_FDLSYSTEM
                                                        .EXTRN  ANLRMS$_FDLSOURCE
                                                        .EXTRN  ANLRMS$_FDLFILE
                                                        .EXTRN  ANLRMS$_FDLALLOC
                                                        .EXTRN  ANLRMS$_FDLNOALLOC
                                                        .EXTRN  ANLRMS$_FDLBESTTRY
                                                        .EXTRN  ANLRMS$_FDLBUCKETSIZE
                                                        .EXTRN  ANLRMS$_FDLCLUSTERSIZE
                                                        .EXTRN  ANLRMS$_FDLCONTIG
                                                        .EXTRN  ANLRMS$_FDLEXTENSION
                                                        .EXTRN  ANLRMS$_FDLGLOBALBUFS
```

```
                                                        .EXTRN  ANLRMS$_FDLMAXRECORD
                                                        .EXTRN  ANLRMS$_FDLFILENAME
                                                        .EXTRN  ANLRMS$_FDLORG, ANLRMS$_FDLOWNER
                                                        .EXTRN  ANLRMS$_FDLPROTECTION
                                                        .EXTRN  ANLRMS$_FDLRECORD
                                                        .EXTRN  ANLRMS$_FDLSPAN
                                                        .EXTRN  ANLRMS$_FDLCC, ANLRMS$_FDLVFCSIZE
                                                        .EXTRN  ANLRMS$_FDLFORMAT
                                                        .EXTRN  ANLRMS$_FDLSIZE
                                                        .EXTRN  ANLRMS$_FDLAREA
                                                        .EXTRN  ANLRMS$_FDLKEY, ANLRMS$_FDLCHANGES
                                                        .EXTRN  ANLRMS$_FDLDATAAREA
                                                        .EXTRN  ANLRMS$_FDLDATAFILL
                                                        .EXTRN  ANLRMS$_FDLDATAKEYCOMPB
                                                        .EXTRN  ANLRMS$_FDLDATARECCOMPB
                                                        .EXTRN  ANLRMS$_FDLDUPS
                                                        .EXTRN  ANLRMS$_FDLINDEXAREA
                                                        .EXTRN  ANLRMS$_FDLINDEXCOMPB
                                                        .EXTRN  ANLRMS$_FDLINDEXFILL
                                                        .EXTRN  ANLRMS$_FDLL1INDEXAREA
                                                        .EXTRN  ANLRMS$_FDLKEYNAME
                                                        .EXTRN  ANLRMS$_FDLNORECS
                                                        .EXTRN  ANLRMS$_FDLNULLKEY
                                                        .EXTRN  ANLRMS$_FDLNULLVALUE
                                                        .EXTRN  ANLRMS$_FDLPROLOG
                                                        .EXTRN  ANLRMS$_FDLSEGLENGTH
                                                        .EXTRN  ANLRMS$_FDLSEGPOS
                                                        .EXTRN  ANLRMS$_FDLSEGTYPE
                                                        .EXTRN  ANLRMS$_FDLANALAREA
                                                        .EXTRN  ANLRMS$_FDLRECL
                                                        .EXTRN  ANLRMS$_FDLANALKEY
                                                        .EXTRN  ANLRMS$_FDLDATAKEYCOMP
                                                        .EXTRN  ANLRMS$_FDLDATARECCOMP
                                                        .EXTRN  ANLRMS$_FDLDATARECS
                                                        .EXTRN  ANLRMS$_FDLDATASPACE
                                                        .EXTRN  ANLRMS$_FDLDEPTH
                                                        .EXTRN  ANLRMS$_FDLDUPSPER
                                                        .EXTRN  ANLRMS$_FDLIDXCOMP
                                                        .EXTRN  ANLRMS$_FDLIDXFILL
                                                        .EXTRN  ANLRMS$_FDLIDXSPACE
                                                        .EXTRN  ANLRMS$_FDLIDXL1RECS
                                                        .EXTRN  ANLRMS$_FDLDATALENMEAN
                                                        .EXTRN  ANLRMS$_FDLIDXLENMEAN
                                                        .EXTRN  ANLRMS$_STATAREA
                                                        .EXTRN  ANLRMS$_STATRECL
                                                        .EXTRN  ANLRMS$_STATKEY
                                                        .EXTRN  ANLRMS$_STATDEPTH
                                                        .EXTRN  ANLRMS$_STATIDXL1RECS
                                                        .EXTRN  ANLRMS$_STATIDXLENMEAN
                                                        .EXTRN  ANLRMS$_STATIDXSPACE
                                                        .EXTRN  ANLRMS$_STATIDXFILL
                                                        .EXTRN  ANLRMS$_STATIDXCOMP
                                                        .EXTRN  ANLRMS$_STATDATARECS
                                                        .EXTRN  ANLRMS$_STATDUPSPER
                                                        .EXTRN  ANLRMS$_STATDATALENMEAN
                                                        .EXTRN  ANLRMS$_STATDATASPACE
                                                        .EXTRN  ANLRMS$_STATDATAFILL
```

```
.EXTRN  ANLRMS$_STATDATAKEYCOMP
.EXTRN  ANLRMS$_STATDATARECCOMP
.EXTRN  ANLRMS$_STATEFFICIENCY
.EXTRN  ANLRMS$_BADAREA1ST2
.EXTRN  ANLRMS$_BADAREABKTSIZE
.EXTRN  ANLRMS$_BADAREAFIT
.EXTRN  ANLRMS$_BADAREAID
.EXTRN  ANLRMS$_BADAREANEXT
.EXTRN  ANLRMS$_BADAREAROOT
.EXTRN  ANLRMS$_BADAREAUSED
.EXTRN  ANLRMS$_BADBKTAREAID
.EXTRN  ANLRMS$_BADBKTCHECK
.EXTRN  ANLRMS$_BADBKTFREE
.EXTRN  ANLRMS$_BADBKTKEYID
.EXTRN  ANLRMS$_BADBKTLEVEL
.EXTRN  ANLRMS$_BADBKTROOTBIT
.EXTRN  ANLRMS$_BADBKTSAMPLE
.EXTRN  ANLRMS$_BADCELLFIT
.EXTRN  ANLRMS$_BADCHECKSUM
.EXTRN  ANLRMS$_BADDATARECBITS
.EXTRN  ANLRMS$_BADDATARECFIT
.EXTRN  ANLRMS$_BADDATARECPS
.EXTRN  ANLRMS$_BAD3IDXKEYFIT
.EXTRN  ANLRMS$_BADIDXLASTKEY
.EXTRN  ANLRMS$_BADIDXORDER
.EXTRN  ANLRMS$_BADIDXRECBITS
.EXTRN  ANLRMS$_BADIDXRECFIT
.EXTRN  ANLRMS$_BADIDXRECPS
.EXTRN  ANLRMS$_BADKEYAREAID
.EXTRN  ANLRMS$_BADKEYDATABKT
.EXTRN  ANLRMS$_BADKEYDATAFIT
.EXTRN  ANLRMS$_BADKEYDATATYPE
.EXTRN  ANLRMS$_BADKEYIDXBKT
.EXTRN  ANLRMS$_BADKEYFILL
.EXTRN  ANLRMS$_BADKEYFIT
.EXTRN  ANLRMS$_BADKEYREFID
.EXTRN  ANLRMS$_BADKEYROOTLEVEL
.EXTRN  ANLRMS$_BADKEYSEGCOUNT
.EXTRN  ANLRMS$_BADKEYSEGVEC
.EXTRN  ANLRMS$_BADKEYSUMMARY
.EXTRN  ANLRMS$_BADREADNOPAR
.EXTRN  ANLRMS$_BADREADPAR
.EXTRN  ANLRMS$_BADSIDRDUPCT
.EXTRN  ANLRMS$_BADSIDRPTRFIT
.EXTRN  ANLRMS$_BADSIDRPTRSZ
.EXTRN  ANLRMS$_BADSIDRSIZE
.EXTRN  ANLRMS$_BADSTREAMEOF
.EXTRN  ANLRMS$_BADVBNFREE
.EXTRN  ANLRMS$_BKTLOOP
.EXTRN  ANLRMS$_EXTENDERR
.EXTRN  ANLRMS$_FLAGERROR
.EXTRN  ANLRMS$_MISSINGBKT
.EXTRN  ANLRMS$_NOTOK, ANLRMS$_SPANERROR
.EXTRN  ANLRMS$_TOOMANYRECS
.EXTRN  ANLRMS$_UNWIND, ANLRMS$_VFCTOOSHORT
.EXTRN  ANLRMS$_CACHEFULL
.EXTRN  ANLRMS$_CACHERELFAIL
```

```
                                                         .EXTRN  ANLRMS$_FACILITY
                                                         .EXTRN  CLI$GET_VALUE, CLI$PRESENT
                                                         .EXTRN  LIB$LP_LINES, LIB$PUT_OUTPUT
                                                         .EXTRN  STR$TRIM, ANL$GB_MODE
                                                         .EXTRN  ANL$GL_FAT, SYS$PARSE
                                                         .EXTRN  SYS$SEARCH, SYS$CREATE
                                                         .EXTRN  SYS$CONNECT

                                                         .PSECT  $CODE$,NOWRT,2

                                        03FC 00000       .ENTRY  ANL$PREPARE_REPORT_FILE, Save R2,R3,R4,R5,- ; 0702
                                                                 R6,R7,R8,R9                                 ;
                    59      0000'  CF  9E 00002          MOVAB   P.AAA, R9                                   ;
                    58  00000000G  00  9E 00007          MOVAB   LIB$SIGNAL, R8                              ;
                    57      0000'  CF  9E 0000E          MOVAB   REPORT_FILE_SPEC, R7                        ;
                    56        08   AC  D0 00013          MOVL    INPUT_SPEC, R6                              ; 0705
              01A4  C7             66  B0 00017          MOVW    (R6), INPUT_FILE_SPEC                       ; 0717
      01A8   D7       04  B6       66  28 0001C          MOVC3   (R6), @4(R6), @INPUT_FILE_SPEC+4            ; 0718
                                   59  DD 00023          PUSHL   R9                                          ; 0722
         00000000G  00             01  FB 00025          CALLS   #1, CLI$PRESENT                             ;
              019C  C7             50  90 0002C          MOVB    R0, GENERATING_REPORT                       ;
                    01      019C   C7  E8 00031          BLBS    GENERATING_REPORT, 1$                       ; 0723
                                   04  00036             RET                                                 ;
                    03      02BC   C7  E8 00037 1$:      BLBS    FIRST_CALL, 2$                              ; 0729
                                 0125  31 0003C          BRW     14$                                         ;
                                   57  DD 0003F 2$:      PUSHL   R7                                          ; 0735
                              10   A9  9F 00041          PUSHAB  P.AAC                                       ;
         00000000G  00             02  FB 00044          CALLS   #2, CLI$GET_VALUE                           ;
              02B4  C7             50  90 0004B          MOVB    R0, REPORT_TO_FILE                          ;
                                   57  DD 00050          PUSHL   R7                                          ; 0736
                                   57  DD 00052          PUSHL   R7                                          ;
                                   57  DD 00054          PUSHL   R7                                          ;
         00000000G  00             03  FB 00056          CALLS   #3, STR$TRIM                                ;
                    50      0000G  CF  9A 0005D          MOVZBL  ANL$GB_MODE, R0                             ; 0740
                    01             50  91 00062          CMPB    R0, #1                                      ; 0742
                                   0A  13 00065          BEQL    3$                                          ;
                    04             50  91 00067          CMPB    R0, #4                                      ;
                                   2B  1F 0006A          BLSSU   5$                                          ;
                    05             50  91 0006C          CMPB    R0, #5                                      ;
                                   26  1A 0006F          BGTRU   5$                                          ;
                    14      02B4   C7  E9 00071 3$:      BLBC    REPORT_TO_FILE, 4$                          ; 0751
              0134  C7        04   A7  D0 00076          MOVL    REPORT_FILE_SPEC+4, REPORT_FAB+44           ; 0752
              013C  C7             67  90 0007C          MOVB    REPORT_FILE_SPEC, REPORT_FAB+52             ; 0753
              0138  C7        18   A9  9E 00081          MOVAB   P.AAE, REPORT_FAB+48                        ; 0754
                                 0089  31 00087          BRW     10$                                         ; 0755
              0134  C7        23   A9  9E 0008A 4$:      MOVAB   P.AAF, REPORT_FAB+44                        ; 0757
              013C  C7             0A  90 00090          MOVB    #10, REPORT_FAB+52                          ; 0758
                                   5F  11 00095          BRB     8$                                          ; 0751
                    02             50  91 00097 5$:      CMPB    R0, #2                                      ; 0761
                                   5C  12 0009A          BNEQ    9$                                          ;
              0134  C7        04   A6  D0 0009C          MOVL    4(R6), REPORT_FAB+44                        ; 0773
              013C  C7             66  90 000A2          MOVB    (R6), REPORT_FAB+52                         ; 0774
              0130  C7      FF40   C7  9E 000A7          MOVAB   RELATED_NAM, REPORT_FAB+40                  ; 0775
                            0108   C7  9F 000AE          PUSHAB  REPORT_FAB                                  ; 0776
         00000000G  00             01  FB 000B2          CALLS   #1, SYS$PARSE                               ;
                    52             50  D0 000B9          MOVL    R0, STATUS                                  ;
                    05             52  E8 000BC          BLBS    STATUS, 6$                                  ; 0777
```

```
                                  52  DD 000BF        PUSHL   STATUS                                   ;
                   6B             01  FB 000C1        CALLS   #1, LIB$SIGNAL                           ;
                          0108    C7  9F 000C4 6$:    PUSHAB  REPORT_FAB                               ; 0778
         00000000G  00            01  FB 000C8        CALLS   #1, SYS$SEARCH                           ;
                    52            50  D0 000CF        MOVL    R0, STATUS                               ;
                    05            52  E8 000D2        BLBS    STATUS, 7$                               ; 0779
                                  52  DD 000D5        PUSHL   STATUS                                   ;
                    68            01  FB 000D7        CALLS   #1, LIB$SIGNAL                           ;
             0134   C7      04    A7  D0 000DA 7$:    MOVL    REPORT_FILE_SPEC+4, REPORT_FAB+44        ; 0784
             013C   C7            67  90 000E0        MOVB    REPORT_FILE_SPEC, REPORT_FAB+52          ; 0785
             0138   C7      2D    A9  9E 000E5        MOVAB   P.AAG, REPORT_FAB+48                     ; 0786
             013D   C7            04  90 000EB        MOVB    #4, REPORT_FAB+53                        ; 0787
             0130   C7      A0    A7  9E 000F0        MOVAB   REPORT_NAM, REPORT_FAB+40                ; 0788
                                  20  11 000F6 8$:    BRB     11$                                      ; 0740
                    03            50  91 000F8 9$:    CMPB    R0, #3                                   ; 0790
                                  1B  12 000FB        BNEQ    11$                                      ;
                    6B    02B4    C7  E9 000FD        BLBC    REPORT_TO_FILE, 15$                      ; 0797
             0134   C7      04    A7  D0 00102        MOVL    REPORT_FILE_SPEC+4, REPORT_FAB+44        ; 0798
             013C   C7            67  90 00108        MOVB    REPORT_FILE_SPEC, REPORT_FAB+52          ; 0799
             0138   C7      31    A9  9E 0010D        MOVAB   P.AAH, REPORT_FAB+48                     ; 0800
             013D   C7            0B  90 00113 10$:   MOVB    #11, REPORT_FAB+53                       ; 0801
                          0108    C7  9F 00118 11$:   PUSHAB  REPORT_FAB                               ; 0809
         00000000G  00            01  FB 0011C        CALLS   #1, SYS$CREATE                           ;
                    52            50  D0 00123        MOVL    R0, STATUS                               ;
             FC38   C7      AB    A7  9B 00126        MOVZBW  REPORT_NAM+11, EXPANDED_SPEC             ; 0810
                    15            52  E8 0012C        BLBS    STATUS, 12$                              ; 0811
                          0114    C7  DD 0012F        PUSHL   REPORT_FAB+12                            ;
                                  52  DD 00133        PUSHL   STATUS                                   ;
                          FC38    C7  9F 00135        PUSHAB  EXPANDED_SPEC                            ;
                                  01  DD 00139        PUSHL   #1                                       ;
                      00B110A4    8F  DD 0013B        PUSHL   #11604132                                ;
                    68            05  FB 00141        CALLS   #5, LIB$SIGNAL                           ;
                          0158    C7  9F 00144 12$:   PUSHAB  REPORT_RAB                               ; 0812
         00000000G  00            01  FB 00148        CALLS   #1, SYS$CONNECT                          ;
                    52            50  D0 0014F        MOVL    R0, STATUS                               ;
                    05            52  E8 00152        BLBS    STATUS, 13$                              ; 0813
                                  52  DD 00155        PUSHL   STATUS                                   ;
                    68            01  FB 00157        CALLS   #1, LIB$SIGNAL                           ;
             01A0   C7      04    AC  D0 0015A 13$:   MOVL    HEADING_MSG, REPORT_HEADING_MSG          ; 0817
                          02BC    C7  94 00160        CLRB    FIRST_CALL                               ; 0819
                          02AC    C7  D4 00164 14$:   CLRL    PAGE_NUMBER                              ; 0824
             0000V  CF            00  FB 00168        CALLS   #0, ANL$REPORT_PAGE                      ; 0825
                                      04 0016D 15$:   RET                                              ; 0829

; Routine Size:  366 bytes,    Routine Base:  $CODE$ + 0000
```

```
1 %sbttl 'ANL$REPORT_PAGE - Eject Page in Report'
1 !++
1 ! Functional Description:
1 !       This routine is called to eject the page in a report and print
1 !       the heading on the new page.
1 !
1 ! Formal Parameters:
1 !       none
1 !
1 ! Implicit Inputs:
1 !       global data
1 !
1 ! Implicit Outputs:
1 !       global data
1 !
1 ! Returned Value:
1 !       none
1 !
1 ! Side Effects:
1 !
1 !--
1
1
2 global routine anl$report_page: novalue = begin
2
2
2 ! Since we are starting a new page, reset the line counter.
2
2 line_counter = lib$lp_lines() - 7;
2
2 ! Now we can eject and print the heading line.  Don't do this if the
2 ! current heading message is zero - page headers are not desired.
2
3 if .report_heading_msg nequ 0 then (
3         anl$format_line(-1,0,anlrms$_anything,describe(%char(formfeed)));
3
3         increment (page_number);
3         anl$format_line(-1,0,.report_heading_msg,0,.page_number);
3         anl$format_line(-1,0,anlrms$_anything,input_file_spec);
3         anl$format_skip(-1);
3         anl$format_skip(-1);
2 );
2
2 return;
2
1 end;
```

```
                                        .PSECT  $PLIT$,NOWRT,NOEXE,2

              0C  00044 P.AAJ:          .ASCII  <12>                    ;
                  00045                 .BLKB   3
        00000001  00048 P.AAI:          .LONG   1                       ;
        00000000' 0004C                 .ADDRESS P.AAJ                  ;
```

```
                                                            .PSECT  $CODE$,NOWRT,2

                                         000C 00000         .ENTRY  ANL$REPORT_PAGE, Save R2,R3                 ; 0853
                   53      0000V    CF    9E 00002          MOVAB   ANL$FORMAT_LINE, R3                         ;
                   52  00000000G    8F    D0 00007          MOVL    #ANLRMS$_ANYTHING, R2                       ;
    00000000G      00               00    FB 0000E          CALLS   #0, LIB$LP_LINES                            ; 0858
        0000'      CF         F9    A0    9E 00015          MOVAB   -7(R0), LINE_COUNTER                        ;
                           0000'    CF    D5 0001B          TSTL    REPORT_HEADING_MSG                          ; 0863
                                    42    13 0001F          BEQL    1$                                          ;
                           0000'    CF    9F 00021          PUSHAB  P.AAI                                       ; 0864
                                    52    DD 00025          PUSHL   R2                                          ;
                                    7E    D4 00027          CLRL    -(SP)                                       ;
                   7E               01    CE 00029          MNEGL   #1, -(SP)                                   ;
                   63               04    FB 0002C          CALLS   #4, ANL$FORMAT_LINE                         ;
                           0000'    CF    D6 0002F          INCL    PAGE_NUMBER                                 ; 0866
                           0000'    CF    DD 00033          PUSHL   PAGE_NUMBER                                 ; 0867
                                    7E    D4 00037          CLRL    -(SP)                                       ;
                           0000'    CF    DD 00039          PUSHL   REPORT_HEADING_MSG                          ;
                                    7E    D4 0003D          CLRL    -(SP)                                       ;
                   7E               01    CE 0003F          MNEGL   #1, -(SP)                                   ;
                   63               05    FB 00042          CALLS   #5, ANL$FORMAT_LINE                         ;
                           0000'    CF    9F 00045          PUSHAB  INPUT_FILE_SPEC                             ; 0868
                                    52    DD 00049          PUSHL   R2                                          ;
                                    7E    D4 0004B          CLRL    -(SP)                                       ;
                   7E               01    CE 0004D          MNEGL   #1, -(SP)                                   ;
                   63               04    FB 00050          CALLS   #4, ANL$FORMAT_LINE                         ;
                   7E               01    CE 00053          MNEGL   #1, -(SP)                                   ; 0869
        0000V      CF               01    FB 00056          CALLS   #1, ANL$FORMAT_SKIP                         ;
                   7E               01    CE 0005B          MNEGL   #1, -(SP)                                   ; 0870
        0000V      CF               01    FB 0005E          CALLS   #1, ANL$FORMAT_SKIP                         ;
                                          04 00063 1$:      RET                                                 ; 0875

; Routine Size:  100 bytes,    Routine Base:  $CODE$ + 016E
```

```
;   372          0876  1 %sbttl 'ANL$FORMAT_LINE - Format a Line of the Report'
;   373          0877  1 !++
;   374          0878  1 ! Functional Description:
;   375          0879  1 !       This routine is called to format a line and place it in the current
;   376          0880  1 !       report file.  It also goes to the terminal if this is an interactive
;   377          0881  1 !       session.
;   378          0882  1 !
;   379          0883  1 ! Formal Parameters:
;   380          0884  1 !       widow_control   Controls widowing as follows:
;   381          0885  1 !                               positive        specifies number of lines that
;   382          0886  1 !                                               must remain on the page.
;   383          0887  1 !                               zero            doesn't matter how many lines.
;   384          0888  1 !                               negative        Same as zero, but don't send
;   385          0889  1 !                                               the line to the screen.
;   386          0890  1 !       indent_level    The number of tab stops to indent the line.
;   387          0891  1 !       template_msg    The status code of the message defining the line
;   388          0892  1 !                       template.
;   389          0893  1 !       fao1...         $FAO arguments to fill into the message.
;   390          0894  1 !
;   391          0895  1 ! Implicit Inputs:
;   392          0896  1 !       global data
;   393          0897  1 !
;   394          0898  1 ! Implicit Outputs:
;   395          0899  1 !       global data
;   396          0900  1 !
;   397          0901  1 ! Returned Value:
;   398          0902  1 !       none
;   399          0903  1 !
;   400          0904  1 ! Side Effects:
;   401          0905  1 !
;   402          0906  1 !--
;   403          0907  1
;   404          0908  1
;   405          0909  2 global routine anl$format_line(widow_control,indent_level,template_msg,fao1): novalue = begin
;   406          0910  2
;   407          0911  2 local
;   408          0912  2         status: long;
;   409          0913  2
;   410          0914  2
;   411          0915  2 ! If we aren't generating a report, then drop this line in the bit bucket.
;   412          0916  2
;   413          0917  2 if not .generating_report then
;   414          0918  2         return;
;   415          0919  2
;   416          0920  2 ! First we obtain the text of the template message.
;   417          0921  2
;   418          0922  3 begin
;   419          0923  3 local
;   420          0924  3         local_described_buffer(template_buf,nam$c_maxrss);
;   421          0925  3
;   422        P 0926  3 status = $getmsg(msgid=.template_msg,
;   423        P 0927  3                  msglen=template_buf,
;   424        P 0928  3                  bufadr=template_buf,
;   425          0929  3                  flags=%b'0001');
;   426          0930  3 check (.status,.status);
;   427          0931  3
;   428          0932  3 ! Now we can plug the $FAO arguments into the message template.
```

```
;  429          0933  3
;  430          0934  4 begin
;  431          0935  4 local
;  432          0936  4         local_described_buffer(result_buf,132);
;  433          0937  4
;  434        P 0938  4 status = $faol(ctrstr=template_buf,
;  435        P 0939  4                outlen=result_buf,
;  436        P 0940  4                outbuf=result_buf,
;  437          0941  4                prmlst=faol);
;  438          0942  4 check (.status,.status);
;  439          0943  4
;  440          0944  4 ! Prefix the resulting text with enough tabs to effect the indentation.
;  441          0945  4
;  442          0946  4 ch$move(.result_buf[len],.result_buf[ptr], .result_buf[ptr]+.indent_level);
;  443          0947  4 result_buf[len] = .result_buf[len] + .indent_level;
;  444          0948  4 ch$fill(%char(tab), .indent_level,.result_buf[ptr]);
;  445          0949  4
;  446          0950  4 ! There are two cases for widow control.  If zero, then only eject if we
;  447          0951  4 ! are out of lines.  If positive, then eject if there are not said number
;  448          0952  4 ! of lines left on the page.
;  449          0953  4
;  450          0954  4 if (.widow_control leq 0 and .line_counter leq 0) or
;  451          0955  4    (.widow_control geq 1 and .line_counter lss .widow_control) then
;  452          0956  4         anl$report_page();
;  453          0957  4
;  454          0958  4 ! If there is a current report file, put the line into it.  Also account
;  455          0959  4 ! for the line on the page.
;  456          0960  4
;  457          0961  5 if .report_rab[rab$w_isi] negu 0 then (
;  458          0962  5         report_rab[rab$w_rsz] = .result_buf[len];
;  459          0963  5         report_rab[rab$l_rbf] = .result_buf[ptr];
;  460          0964  5         status = $put(rab=report_rab);
;  461          0965  5         check (.status, anlrms$_writeerr,1,report_file_spec,.status,.report_rab[rab$l_stv]);
;  462          0966  5         decrement (line_counter);
;  463          0967  4 );
;  464          0968  4
;  465          0969  4 ! If we are doing an interactive session, also put the line onto the screen.
;  466          0970  4 ! However, lines with widow control of -1 are not displayed.
;  467          0971  4
;  468          0972  5 if .anl$gb_mode eqlu anl$k_interactive and .widow_control geq 0 then (
;  469          0973  5         status = lib$put_output(result_buf);
;  470          0974  5         check (.status, .status);
;  471          0975  4 );
;  472          0976  4
;  473          0977  3 end;
;  474          0978  2 end;
;  475          0979  2
;  476          0980  2 return;
;  477          0981  2
;  478          0982  1 end;


                                                        .EXTRN  SYS$GETMSG, SYS$FAOL
                                                        .EXTRN  SYS$PUT

                                        01FC 00000      .ENTRY  ANL$FORMAT_LINE, Save R2,R3,R4,R5,R6,R7,R8  ; 0909
```

```
                                58 00000000G   00   9E 00002          MOVAB   LIB$SIGNAL, R8                                    ;
                                57      0000'  CF   9E 00009          MOVAB   LINE_COUNTER, R7                                  ;
                                5E      FE6C   CE   9E 0000E          MOVAB   -404(SP), SP                                      ;
                                01      FEEC   C7   E8 00013          BLBS    GENERATING_REPORT, 1$                             ; 0917
                                               04   00018          RET                                                       ;
                           008C CE        FF   8F   9A 00019 1$:     MOVZBL  #255, TEMPLATE_BUF                                ; 0924
                           0090 CE      0094   CE   9E 0001F          MOVAB   TEMPLATE_BUF+8, TEMPLATE_BUF+4                    ;
                                7E              01   7D 00026          MOVQ    #1, -(SP)                                         ; 0929
                                            0094 CE   9F 00029          PUSHAB  TEMPLATE_BUF                                      ;
                                            0098 CE   9F 0002D          PUSHAB  TEMPLATE_BUF                                      ;
                                              0C AC   DD 00031          PUSHL   TEMPLATE_MSG                                      ;
                        00000000G 00            05   FB 00034          CALLS   #5, SYS$GETMSG                                    ;
                                56              50   D0 0003B          MOVL    R0, STATUS                                        ;
                                05              56   E8 0003E          BLBS    STATUS, 2$                                        ; 0930
                                                56   DD 00041          PUSHL   STATUS                                            ;
                                68              01   FB 00043          CALLS   #1, LIB$SIGNAL                                    ;
                                6E        84    8F   9A 00046 2$:     MOVZBL  #132, RESULT_BUF                                  ; 0936
                           04   AE        08   AE   9E 0004A          MOVAB   RESULT_BUF+8, RESULT_BUF+4                        ;
                                            10   AC   9F 0004F          PUSHAB  FAO1                                              ; 0941
                                            04   AE   9F 00052          PUSHAB  RESULT_BUF                                        ;
                                            08   AE   9F 00055          PUSHAB  RESULT_BUF                                        ;
                                          0098   CE   9F 00058          PUSHAB  TEMPLATE_BUF                                      ;
                        00000000G 00            04   FB 0005C          CALLS   #4, SYS$FAOL                                      ;
                                56              50   D0 00063          MOVL    R0, STATUS                                        ;
                                05              56   E8 00066          BLBS    STATUS, 3$                                        ; 0942
                                                56   DD 00069          PUSHL   STATUS                                            ;
                                68              01   FB 0006B          CALLS   #1, LIB$SIGNAL                                    ;
                     50         04   AE        08   AC   C1 0006E 3$:     ADDL3   INDENT_LEVEL, RESULT_BUF+4, R0                    ; 0946
                     60         04   BE             6E   28 00074          MOVC3   RESULT_BUF, @RESULT_BUF+4, (R0)                   ;
                                            6E        08   AC   A0 00079          ADDW2   INDENT_LEVEL, RESULT_BUF                  ; 0947
   08   AC                  09            6E              00   2C 0007D          MOVC5   #0, (SP), #9, INDENT_LEVEL, @RESULT_BUF+4 ; 0948
                                                04   BE   00083                                                                  ;
                                52        04   AC   D0 00085          MOVL    WIDOW_CONTROL, R2                                 ; 0954
                                                04   14 00089          BGTR    4$                                                ;
                                                67   D5 0008B          TSTL    LINE_COUNTER                                      ;
                                                09   15 0008D          BLEQ    5$                                                ;
                                                52   D5 0008F 4$:     TSTL    R2                                                ; 0955
                                                0A   15 00091          BLEQ    6$                                                ;
                                52              67   D1 00093          CMPL    LINE_COUNTER, R2                                  ;
                                                05   18 00096          BGEQ    6$                                                ;
                           FEFF CF              00   FB 00098 5$:     CALLS   #0, ANL$REPORT_PAGE                               ; 0956
                                      FEAA      C7   B5 0009D 6$:     TSTW    REPORT_RAB+2                                      ; 0961
                                                33   13 000A1          BEQL    8$                                                ;
                           FECA C7              6E   B0 000A3          MOVW    RESULT_BUF, REPORT_RAB+34                         ; 0962
                           FED0 C7        04    AE   D0 000A8          MOVL    RESULT_BUF+4, REPORT_RAB+40                       ; 0963
                                      FEA8      C7   9F 000AE          PUSHAB  REPORT_RAB                                        ; 0964
                        00000000G 00            01   FB 000B2          CALLS   #1, SY$PUT                                        ;
                                56              50   D0 000B9          MOVL    R0, STATUS                                        ;
                                15              56   E8 000BC          BLBS    STATUS, 7$                                        ; 0965
                                      FEB4      C7   DD 000BF          PUSHL   REPORT_RAB+12                                     ;
                                                56   DD 000C3          PUSHL   STATUS                                            ;
                                      FD50      C7   9F 000C5          PUSHAB  REPORT_FILE_SPEC                                  ;
                                                01   DD 000C9          PUSHL   #1                                                ;
                                00B110D4        8F   DD 000CB          PUSHL   #11604180                                         ;
                                68              05   FB 000D1          CALLS   #5, LIB$SIGNAL                                    ;
                                                67   D7 000D4 7$:     DECL    LINE_COUNTER                                      ; 0966
                                03      0000G   CF   91 000D6 8$:     CMPB    ANL$GB_MODE, #3                                   ; 0972
```

```
                                          18  12 000DB          BNEQ    9$                                                      ;
                                          52  D5 000DD          TSTL    R2                                                      ;
                                          14  19 000DF          BLSS    9$                                                      ;
                                          5E  DD 000E1          PUSHL   SP                                                      ; 0973
                          00000000G   00  01  FB 000E3          CALLS   #1, LIB$PUT_OUTPUT                                      ;
                                      56  50  D0 000EA          MOVL    R0, STATUS                                              ;
                                      05  56  E8 000ED          BLBS    STATUS, 9$                                              ; 0974
                                          56  DD 000F0          PUSHL   STATUS                                                  ;
                                      68  01  FB 000F2          CALLS   #1, LIB$SIGNAL                                          ;
                                          04  000F5 9$:          RET                                                             ; 0982

; Routine Size:  246 bytes,    Routine Base:  $CODE$ + 01D2
```

```
1 %sbttl 'ANL$FORMAT_SKIP - Skip a Line in Report'
1 !++
1 ! Functional Description:
1 !       This routine can be called to skip a line in the current report.
1 !
1 ! Formal Parameters:
1 !       widow_control   See ANL$FORMAT_LINE
1 !
1 ! Implicit Inputs:
1 !       global data
1 !
1 ! Implicit Outputs:
1 !       global data
1 !
1 ! Returned Value:
1 !       none
1 !
1 ! Side Effects:
1 !
1 !--
1
1
2 global routine anl$format_skip(widow_control): novalue = begin
2
2
2 ! Just call FORMAT_LINE with a blank line.
2
2 anl$format_line(.widow_control,0,anlrms$_anything,describe(''));
2
2 return;
2
1 end;


                                                .PSECT  $PLIT$,NOWRT,NOEXE,2

                         00050 P.AAL:   .BLKB   0                                   ;
               00000000  00050 P.AAK:   .LONG   0                                   ;
               00000000' 00054          .ADDRESS P.AAL                              ;


                                                .PSECT  $CODE$,NOWRT,2

                     0000 00000         .ENTRY  ANL$FORMAT_SKIP, Save nothing      ; 1005
            0000'  CF  9F 00002         PUSHAB  P.AAK                               ; 1010
        00000000G  8F  DD 00006         PUSHL   #ANLRMS$_ANYTHING                   ;
                   7E  D4 0000C         CLRL    -(SP)                               ;
               04  AC  DD 0000E         PUSHL   WIDOW_CONTROL                       ;
   FEF4   CF       04  FB 00011         CALLS   #4, ANL$FORMAT_LINE                 ;
                       04 00016         RET                                         ; 1014

; Routine Size:  23 bytes,    Routine Base:  $CODE$ + 02C8
```

```
;  513          1015  1 %sbttl 'ANL$FORMAT_ERROR - Put Error Message in Report'
;  514          1016  1 !++
;  515          1017  1 ! Functional Description:
;  516          1018  1 !       This routine is called to format an error message into the report
;  517          1019  1 !       file.
;  518          1020  1 !
;  519          1021  1 ! Formal Parameters:
;  520          1022  1 !       error_msg       Status code for the error message.
;  521          1023  1 !       fao1...         $FAO substitution parameters for the message.
;  522          1024  1 !
;  523          1025  1 ! Implicit Inputs:
;  524          1026  1 !       global data
;  525          1027  1 !
;  526          1028  1 ! Implicit Outputs:
;  527          1029  1 !       global data
;  528          1030  1 !
;  529          1031  1 ! Returned Value:
;  530          1032  1 !       none
;  531          1033  1 !
;  532          1034  1 ! Side Effects:
;  533          1035  1 !       anl$worst_error may be set to a new condition value.
;  534          1036  1 !       error_count is incremented.
;  535          1037  1 !
;  536          1038  1 !--
;  537          1039  1
;  538          1040  1
;  539          1041  2 global routine anl$format_error(error_msg,fao1,fao2,fao3,fao4): novalue = begin
;  540          1042  2
;  541          1043  2 external
;  542          1044  2         anl$worst_error;
;  543          1045  2
;  544          1046  2 bind
;  545          1047  2         flag_string = describe('***  ');
;  546          1048  2
;  547          1049  2 builtin
;  548          1050  2         actualcount;
;  549          1051  2
;  550          1052  2 ! We case on the number of $FAO parameters and call ANL$FORMAT_LINE to
;  551          1053  2 ! do the work.  In all cases, however, we add our own first parameter,
;  552          1054  2 ! which is the error message flag string.
;  553          1055  2
;  554          1056  2 case actualcount() from 1 to 5 of set
;  555          1057  2 [1]:    anl$format_line(0,0,,error_msg,flag_string);
;  556          1058  2 [2]:    anl$format_line(0,0,,error_msg,flag_string,,fao1);
;  557          1059  2 [3]:    anl$format_line(0,0,,error_msg,flag_string,,fao1,,fao2);
;  558          1060  2 [4]:    anl$format_line(0,0,,error_msg,flag_string,,fao1,,fao2,,fao3);
;  559          1061  2 [5]:    anl$format_line(0,0,,error_msg,flag_string,,fao1,,fao2,,fao3,,fao4);
;  560          1062  2 tes;
;  561          1063  2
;  562          1064  2 ! Keep track of the number of errors reported.  Also keep track of
;  563          1065  2 ! most severe error which has occurred.
;  564          1066  2
;  565          1067  2 increment (error_count);
;  566          1068  2 if   severity_level (.error_msg) gtr
;  567          1069  3     severity_level (.anl$worst_error)          ! If higher than watermark
;  568          1070  2 then anl$worst_error = .error_msg;           !  -then set new worst error
;  569          1071  2
```

```
;  570          1072  2 return;
;  571          1073  2
;  572          1074  1 end;


                                                                    .PSECT  $PLIT$,NOWRT,NOEXE,2

                                                 20  20  2A  2A  2A  00058 P.AAN:   .ASCII  \***  \                                ;
                                                                     0005D          .BLKB   3
                                                       00000005      00060 P.AAM:   .LONG   5                                        ;
                                                       00000000'     00064          .ADDRESS P.AAN                                   ;

                                                                    FLAG_STRING=        P.AAM
                                                                    .EXTRN  ANL$WORST_ERROR

                                                                    .PSECT  $CODE$,NOWRT,2

                                                          003C      00000          .ENTRY  ANL$FORMAT_ERROR, Save R2,R3,R4,R5       ; 1041
                                                    55    0000'  CF  9E  00002      MOVAB   FLAG_STRING, R5                          ;
                                                    54    FE8    CF  9E  00007      MOVAB   ANL$FORMAT_LINE, R4                      ;
                                                    52      04   AC  D0  0000C      MOVL    ERROR_MSG, R2                            ; 1057
                                          04        01         6C  8F  00010      CASEB   (AP), #1, #4                             ; 1056
                                002C      001F      0013       000A      00014 1$:  .WORD   2$-1$,-                                  ;
                                                               003C      0001C              3$-1$,-                                  ;
                                                                                            4$-1$,-                                  ;
                                                                                            5$-1$,-                                  ;
                                                                                            6$-1$                                    ;
                                                          24  BB  0001E 2$:  PUSHR   #^M<R2,R5>                                   ; 1057
                                                          7E  7C  00020      CLRQ    -(SP)                                        ;
                                                    64        04  FB  00022      CALLS   #4, ANL$FORMAT_LINE                      ;
                                                          38  11  00025      BRB     7$                                           ;
                                                    08    AC  DD  00027 3$:  PUSHL   FAO1                                         ; 1058
                                                          24  BB  0002A      PUSHR   #^M<R2,R5>                                   ;
                                                          7E  7C  0002C      CLRQ    -(SP)                                        ;
                                                    64        05  FB  0002E      CALLS   #5, ANL$FORMAT_LINE                      ;
                                                          2C  11  00031      BRB     7$                                           ;
                                                    7E    08  AC  7D  00033 4$:  MOVQ    FAO1, -(SP)                              ; 1059
                                                          24  BB  00037      PUSHR   #^M<R2,R5>                                   ;
                                                          7E  7C  00039      CLRQ    -(SP)                                        ;
                                                    64        06  FB  0003B      CALLS   #6, ANL$FORMAT_LINE                      ;
                                                          1F  11  0003E      BRB     7$                                           ;
                                                    7E    0C  AC  7D  00040 5$:  MOVQ    FAO2, -(SP)                              ; 1060
                                                          08  AC  DD  00044      PUSHL   FAO1                                     ;
                                                          24  BB  00047      PUSHR   #^M<R2,R5>                                   ;
                                                          7E  7C  00049      CLRQ    -(SP)                                        ;
                                                    64        07  FB  0004B      CALLS   #7, ANL$FORMAT_LINE                      ;
                                                          0F  11  0004E      BRB     7$                                           ;
                                                    7E    10  AC  7D  00050 6$:  MOVQ    FAO3, -(SP)                              ; 1061
                                                    7E    08  AC  7D  00054      MOVQ    FAO1, -(SP)                              ;
                                                          24  BB  00058      PUSHR   #^M<R2,R5>                                   ;
                                                          7E  7C  0005A      CLRQ    -(SP)                                        ;
                                                    64        08  FB  0005C      CALLS   #8, ANL$FORMAT_LINE                      ;
                                                    0000'     CF  D6  0005F 7$:  INCL    ERROR_COUNT                              ; 1067
                                                    50        52  D0  00063      MOVL    R2, TMP_CODE                             ; 1068
                                51      50        03        00  EF  00066      EXTZV   #0, #3, TMP_CODE, R1                     ;
                                50      50        01        00  EF  0006B      EXTZV   #0, #1, TMP_CODE, R0                     ;
```

```
                                 50              04  C4 00070       MULL2   #4, R0                          ;
                                 51              50  C2 00073       SUBL2   R0, R1                          ;
                                 51              03  C0 00076       ADDL2   #3, R1                          ;
                                 50      0000G   CF  D0 00079       MOVL    ANL$WORST_ERROR, TMP_CODE       ; 1069
        53           50          03              00  EF 0007E       EXTZV   #0, #3, TMP_CODE, R3            ;
        50           50          01              00  EF 00083       EXTZV   #0, #1, TMP_CODE, R0            ;
                                 50              04  C4 00088       MULL2   #4, R0                          ;
                                 53              50  C2 0008B       SUBL2   R0, R3                          ;
                                 50       03     A3  9E 0008E       MOVAB   3(R3), R0                       ;
                                 50              51  D1 00092       CMPL    R1, R0                          ;
                                                 05  15 00095       BLEQ    8$                              ;
                      0000G  CF                  52  D0 00097       MOVL    R2, ANL$WORST_ERROR             ; 1070
                                                     04 0009C 8$:   RET                                     ; 1074

; Routine Size:  157 bytes,    Routine Base:  $CODE$ + 02DF
```

```
;  574          1075  1 %sbttl 'ANL$ERROR_COUNT - Report Count of Errors'
;  575          1076  1 !++
;  576          1077  1 ! Functional Description:
;  577          1078  1 !       This routine is called to print a line telling how many errors
;  578          1079  1 !       were discovered during the analysis.
;  579          1080  1 !
;  580          1081  1 ! Formal Parameters:
;  581          1082  1 !       none
;  582          1083  1 !
;  583          1084  1 ! Implicit Inputs:
;  584          1085  1 !       global data
;  585          1086  1 !
;  586          1087  1 ! Implicit Outputs:
;  587          1088  1 !       global data
;  588          1089  1 !
;  589          1090  1 ! Returned Value:
;  590          1091  1 !       none
;  591          1092  1 !
;  592          1093  1 ! Side Effects:
;  593          1094  1 !
;  594          1095  1 !--
;  595          1096  1
;  596          1097  1
;  597          1098  2 global routine anl$error_count: novalue = begin
;  598          1099  2
;  599          1100  2
;  600          1101  2 ! First we print the error count in the report.
;  601          1102  2
;  602          1103  2 if .error_count eqlu 0 then
;  603          1104  2         anl$format_line(0,0,anlrms$_errornone)
;  604          1105  2 else
;  605          1106  2         anl$format_line(0,0,anlrms$_errorcount,.error_count);
;  606          1107  2
;  607          1108  2 ! If this is a /CHECK or /STATISTICS report, we want the user to know
;  608          1109  2 ! what happened.  If the report is going to a file, then we better display
;  609          1110  2 ! a summary line.
;  610          1111  2
;  611          1112  2 if (.anl$gb_mode eqlu anl$k_check or .anl$gb_mode eqlu anl$k_statistics) and
;  612          1113  2    (not .generating_report or .report_to_file)                              then
;  613          1114  2         signal (anlrms$_errors,2,input_file_spec,.error_count);
;  614          1115  2
;  615          1116  2 ! Now we can reset the error counter for the next file.
;  616          1117  2
;  617          1118  2 error_count = 0;
;  618          1119  2
;  619          1120  2 return;
;  620          1121  2
;  621          1122  1 end;


                                0004 00000          .ENTRY  ANL$ERROR_COUNT, Save R2                        ; 1098
                    52  0000'  CF   9E 00002          MOVAB   ERROR_COUNT, R2                                 ;
                    50          62   D0 00007          MOVL    ERROR_COUNT, R0                                 ; 1103
                                0F   12 0000A          BNEQ    1$                                              ;
```

```
                                00000000G   8F   DD 0000C          PUSHL   #ANLRMS$_ERRORNONE                          ; 1104
                                            7E   7C 00012          CLRQ    -(SP)                                      ;
                        FE3D    CF          03   FB 00014          CALLS   #3, ANL$FORMAT_LINE                        ;
                                            0F   11 00019          BRB     2$                                         ;
                                            50   DD 0001B 1$:      PUSHL   R0                                         ; 1106
                                00000000G   8F   DD 0001D          PUSHL   #ANLRMS$_ERRORCOUNT                        ;
                                            7E   7C 00023          CLRQ    -(SP)                                      ;
                        FE2C    CF          04   FB 00025          CALLS   #4, ANL$FORMAT_LINE                        ;
                                01      0000G   CF   91 0002A 2$:      CMPB    ANL$GB_MODE, #1                        ; 1112
                                            07   13 0002F          BEQL    3$                                         ;
                                04      0000G   CF   91 00031          CMPB    ANL$GB_MODE, #4                        ;
                                            1E   12 00036          BNEQ    5$                                         ;
                                04      FEE4    C2   E9 00038 3$:      BLBC    GENERATING_REPORT, 4$                  ; 1113
                                15        FC    A2   E9 0003D          BLBC    REPORT_TO_FILE, 5$                     ;
                                            62   DD 00041 4$:      PUSHL   ERROR_COUNT                                ; 1114
                                        FEEC    C2   9F 00043          PUSHAB  INPUT_FILE_SPEC                        ;
                                            02   DD 00047          PUSHL   #2                                         ;
                                00000000G   8F   DD 00049          PUSHL   #ANLRMS$_ERRORS                            ;
                    00000000G   00              04   FB 0004F          CALLS   #4, LIB$SIGNAL                         ;
                                            62   D4 00056 5$:      CLRL    ERROR_COUNT                                ; 1118
                                                 04 00058          RET                                                ; 1122

; Routine Size:  89 bytes,    Routine Base:  $CODE$ + 037C
```

```
;  623     1123  1 %sbttl 'ANL$FORMAT_FLAGS - Format Flag Bits'
;  624     1124  1 !++
;  625     1125  1 ! Functional Description:
;  626     1126  1 !       This routine is called to format the flags in a byte/word/longword
;  627     1127  1 !       of flags.
;  628     1128  1 !
;  629     1129  1 ! Formal Parameters:
;  630     1130  1 !       indent_level    The level at which the introductory message is to
;  631     1131  1 !                       be indented.  The flags are indented one more level.
;  632     1132  1 !       intro_msg       The introductory message.
;  633     1133  1 !       flags           The flag bits.
;  634     1134  1 !       flag_def        A longword vector defining the flags.  The zeroth
;  635     1135  1 !                       entry specifies the highest-numbered flag.  The
;  636     1136  1 !                       remaining longwords contain the address of a counted
;  637     1137  1 !                       string giving the name of the flag.  If the flag is
;  638     1138  1 !                       undefined, the longword contains zero.
;  639     1139  1 !
;  640     1140  1 ! Implicit Inputs:
;  641     1141  1 !       global data
;  642     1142  1 !
;  643     1143  1 ! Implicit Outputs:
;  644     1144  1 !       global data
;  645     1145  1 !
;  646     1146  1 ! Returned Value:
;  647     1147  1 !       none
;  648     1148  1 !
;  649     1149  1 ! Side Effects:
;  650     1150  1 !
;  651     1151  1 !--
;  652     1152  1
;  653     1153  1
;  654     1154  2 global routine anl$format_flags(indent_level,intro_msg,flags,flag_def): novalue = begin
;  655     1155  2
;  656     1156  2 bind
;  657     1157  2         flags_vector = flags: bitvector[],
;  658     1158  2         flag_def_vector = .flag_def: vector[,long];
;  659     1159  2
;  660     1160  2 local
;  661     1161  2         i: long;
;  662     1162  2
;  663     1163  2
;  664     1164  2 ! Begin by printing the introductory message.
;  665     1165  2
;  666     1166  2 anl$format_line(2,.indent_level,.intro_msg);
;  667     1167  2
;  668     1168  2 ! Now we loop through the flags and process each one that is defined.
;  669     1169  2 ! We print the flag name, bit number, and current setting.
;  670     1170  2
;  671     1171  3 incru i from 0 to .flag_def_vector[0] do (
;  672     1172  3         if .flag_def_vector[.i+1] nequ 0 then
;  673     1173  3                 anl$format_line(0,.indent_level+1,anlrms$_flag,
;  674     1174  3                                 .i,.flag_def_vector[.i+1],.flags_vector[.i]);
;  675     1175  2 );
;  676     1176  2
;  677     1177  2 return;
;  678     1178  2
;  679     1179  1 end;
```

```
                                          0004 00000          .ENTRY  ANL$FORMAT_FLAGS, Save R2                ; 1154
                        7E        04   AC   7D 00002          MOVQ    INDENT_LEVEL, -(SP)                      ; 1166
                                       02   DD 00006          PUSHL   #2                                       ;
                   FDF0 CF             03   FB 00008          CALLS   #3, ANL$FORMAT_LINE                      ;
                                       52   D4 0000D          CLRL    I                                        ; 1171
                                       29   11 0000F          BRB     3$                                       ;
                        50        10 BC42   DE 00011 1$:      MOVAL   @FLAG_DEF[I], R0                         ; 1172
                                  04   A0   D5 00016          TSTL    4(R0)                                    ;
                                       1D   13 00019          BEQL    2$                                       ;
     7E        0C   AC            01   52   EF 0001B          EXTZV   I, #1, FLAGS_VECTOR, -(SP)               ; 1174
                                  04   A0   DD 00021          PUSHL   4(R0)                                    ;
                                       52   DD 00024          PUSHL   I                                        ;
                             00000000G 8F   DD 00026          PUSHL   #ANLRMS$_FLAG                            ; 1173
                   7E        04   AC   01   C1 0002C          ADDL3   #1, INDENT_LEVEL, -(SP)                  ;
                                       7E   D4 00031          CLRL    -(SP)                                    ;
                   FDC5 CF             06   FB 00033          CALLS   #6, ANL$FORMAT_LINE                      ;
                                       52   D6 00038 2$:      INCL    I                                        ; 1171
                     10 BC             52   D1 0003A 3$:      CMPL    I, @FLAG_DEF                             ;
                                       D1   1B 0003E          BLEQU   1$                                       ;
                                            04 00040          RET                                              ; 1179

; Routine Size:  65 bytes,    Routine Base:  $CODE$ + 03D5
```

```
;  681          1180  1 %sbttl 'ANL$FORMAT_HEX - Format Hex Dump of Data'
;  682          1181  1 !++
;  683          1182  1 ! Functional Description:
;  684          1183  1 !       This routine is called to format a hex dump of some bytes.
;  685          1184  1 !       It includes the character representation of the bytes also.
;  686          1185  1 !
;  687          1186  1 ! Formal Parameters:
;  688          1187  1 !       indent_level    The indentation level at which to place the dump.
;  689          1188  1 !       data            Address of descriptor of data to be dumped.
;  690          1189  1 !
;  691          1190  1 ! Implicit Inputs:
;  692          1191  1 !       global data
;  693          1192  1 !
;  694          1193  1 ! Implicit Outputs:
;  695          1194  1 !       global data
;  696          1195  1 !
;  697          1196  1 ! Returned Value:
;  698          1197  1 !       none
;  699          1198  1 !
;  700          1199  1 ! Side Effects:
;  701          1200  1 !
;  702          1201  1 !--
;  703          1202  1
;  704          1203  1
;  705          1204  2 global routine anl$format_hex(indent_level,data): novalue = begin
;  706          1205  2
;  707          1206  2 bind
;  708          1207  2         data_dsc = .data: descriptor,
;  709          1208  2         data_vector = .data_dsc[ptr]: vector[,byte];
;  710          1209  2
;  711          1210  2 local
;  712          1211  2         i: long,
;  713          1212  2         arg_list: vector[20,long],
;  714          1213  2         count: long;
;  715          1214  2
;  716          1215  2 builtin
;  717          1216  2         callg;
;  718          1217  2
;  719          1218  2
;  720          1219  2 ! If the data is null, just quit.
;  721          1220  2
;  722          1221  2 if .data_dsc[len] eqlu 0 then
;  723          1222  2         return;
;  724          1223  2
;  725          1224  2 ! We begin by printing two heading lines.  The first shows the offsets
;  726          1225  2 ! of the bytes and the second is a line of dashes.
;  727          1226  2
;  728          1227  2 anl$format_line(3,.indent_level,anlrms$_hexheading1);
;  729          1228  2 anl$format_line(0,.indent_level,anlrms$_hexheading2);
;  730          1229  2
;  731          1230  2 ! We will be builing argument lists to ANL$FORMAT_LINE.  It will always
;  732          1231  2 ! include widow control, indentation level, and the message code.
;  733          1232  2
;  734          1233  2 arg_list[1] = 0;
;  735          1234  2 arg_list[2] = .indent_level;
;  736          1235  2 arg_list[3] = anlrms$_hexdata;
;  737          1236  2
```

```
;  738          1237  2 ! Now we go into a loop, once through for each 8 bytes to be formatted.
;  739          1238  2
;  740          1239  2 i = 0;
;  741          1240  2 while .i lssu .data_dsc[len] do (
;  742          1241  2
;  743          1242  3         ! Calculate the number of bytes that will go on this line.
;  744          1243  3
;  745          1244  3         count = minu(.data_dsc[len]-.i,8);
;  746          1245  3
;  747          1246  3         ! Next in the argument list we need a count of the spaces to skip
;  748          1247  3         ! so the bytes will be lined up from right to left.
;  749          1248  3
;  750          1249  3         arg_list[4] = (8 - .count) * 3;
;  751          1250  3
;  752          1251  3         ! Now we need the count itself.
;  753          1252  3
;  754          1253  3         arg_list[5] = .count;
;  755          1254  3
;  756          1255  3         ! Now we loop through 8 (or less) bytes and put them in the
;  757          1256  3         ! argument list (backwards, of course).
;  758          1257  3
;  759          1258  3 4       decr j from .count-1 to 0 do (
;  760          1259  3 4               arg_list[6+.j] = .data_vector[.i];
;  761          1260  3 4               increment (i);
;  762          1261  3         );
;  763          1262  3
;  764          1263  3         ! Next we have the byte offset.
;  765          1264  3
;  766          1265  3         arg_list[6+.count] = .i - .count;
;  767          1266  3
;  768          1267  3         ! Now we have to add to the argument list the byte count and a
;  769          1268  3         ! pointer to the byte string.
;  770          1269  3
;  771          1270  3         arg_list[7+.count] = .count;
;  772          1271  3         arg_list[8+.count] = data_vector[.i - .count];
;  773          1272  3
;  774          1273  3         ! Finally, fill in the argument count.
;  775          1274  3
;  776          1275  3         arg_list[0] = 8 + .count;
;  777          1276  3
;  778          1277  3         ! Now we can print the hex data.
;  779          1278  3
;  780          1279  3         callg(arg_list,anl$format_line);
;  781          1280  2 );
;  782          1281  2
;  783          1282  2 return;
;  784          1283  2
;  785          1284  1 end;


                                    003C 00000          .ENTRY  ANL$FORMAT_HEX, Save R2,R3,R4,R5                ; 1204
                              55    FDB6   CF   9E 00002  MOVAB   ANL$FORMAT_LINE, R5                             :
                              5E      B0   AE   9E 00007  MOVAB   -80(SP), SP                                     :
                              54      08   AC   D0 0000B  MOVL    DATA, R4                                        ; 1207
```

```
                                            64  B5 0000F          TSTW     (R4)                                      ; 1221
                                            33  13 00011          BEQL     2$                                        ;
                           00000000G        8F  DD 00013          PUSHL    #ANLRMS$_HEXHEADING1                      ; 1227
                                  04        AC  DD 00019          PUSHL    INDENT_LEVEL                              ;
                                            03  DD 0001C          PUSHL    #3                                        ;
                      65                    03  FB 0001E          CALLS    #3, ANL$FORMAT_LINE                       ;
                           00000000G        8F  DD 00021          PUSHL    #ANLRMS$_HEXHEADING2                      ; 1228
                                  04        AC  DD 00027          PUSHL    INDENT_LEVEL                              ;
                                            7E  D4 0002A          CLRL     -(SP)                                     ;
                      65                    03  FB 0002C          CALLS    #3, ANL$FORMAT_LINE                       ;
                                  04        AE  D4 0002F          CLRL     ARG_LIST+4                                ; 1233
                 08   AE          04        AC  D0 00032          MOVL     INDENT_LEVEL, ARG_LIST+8                  ; 1234
                 0C   AE   00000000G        8F  D0 00037          MOVL     #ANLRMS$_HEXDATA, ARG_LIST+12             ; 1235
                                            53  D4 0003F          CLRL     I                                         ; 1239
      53          64          10            00  ED 00041 1$:      CMPZV    #0, #16, (R4), I                          ; 1240
                                            4F  1B 00046 2$:      BLEQU    6$                                        ;
                      50                    64  3C 00048          MOVZWL   (R4), R0                                  ; 1244
                      50                    53  C2 0004B          SUBL2    I, R0                                     ;
                      08                    50  D1 0004E          CMPL     R0, #8                                    ;
                                            03  1B 00051          BLEQU    3$                                        ;
                      50                    08  D0 00053          MOVL     #8, R0                                    ;
                      52                    50  D0 00056 3$:      MOVL     R0, COUNT                                 ;
                      50          F8        A2  9E 00059          MOVAB    -8(R2), R0                                ; 1249
                      50                    03  C4 0005D          MULL2    #3, R0                                    ;
                 10   AE                    50  CE 00060          MNEGL    R0, ARG_LIST+16                           ;
                 14   AE                    52  D0 00064          MOVL     COUNT, ARG_LIST+20                        ; 1253
                      50                    52  D0 00068          MOVL     COUNT, J                                  ; 1258
                                            09  11 0006B          BRB      5$                                        ;
                 18 AE40          04 B443       9A 0006D 4$:      MOVZBL   @4(R4)[I], ARG_LIST+24[J]                 ; 1259
                                            53  D6 00074          INCL     I                                         ; 1260
                      F4                    50  F4 00076 5$:      SOBGEQ   J, 4$                                     ; 1258
                  50  53                    52  C3 00079          SUBL3    COUNT, I, R0                              ; 1265
                 18 AE42                    50  D0 0007D          MOVL     R0, ARG_LIST+24[COUNT]                    ;
                 1C AE42                    52  D0 00082          MOVL     COUNT, ARG_LIST+28[COUNT]                 ; 1270
                 20 AE42          04 B440       9E 00087          MOVAB    @4(R4)[R0], ARG_LIST+32[COUNT]            ; 1271
                      6E          08        A2  9E 0008E          MOVAB    8(R2), ARG_LIST                           ; 1275
                      65                    6E  FA 00092          CALLG    ARG_LIST, ANL$FORMAT_LINE                 ; 1279
                                            AA  11 00095          BRB      1$                                        ; 1240
                                                04 00097 6$:      RET                                                ; 1284

; Routine Size:  152 bytes,    Routine Base:  $CODE$ + 0416
```

```
;   787          1285  1 %sbttl 'ANL$FORMAT_PROTECTION_MASK - Format Protection Mask'
;   788          1286  1 !++
;   789          1287  1 ! Functional Description:
;   790          1288  1 !       This routine is called to format the standard 16-bit system
;   791          1289  1 !       protection mask.
;   792          1290  1 !
;   793          1291  1 ! Formal Parameters:
;   794          1292  1 !       indent_level    Indentation level in the report.
;   795          1293  1 !       message         Status code for message to use.
;   796          1294  1 !       protection      Protection mask.
;   797          1295  1 !
;   798          1296  1 ! Implicit Inputs:
;   799          1297  1 !       global data
;   800          1298  1 !
;   801          1299  1 ! Implicit Outputs:
;   802          1300  1 !       global data
;   803          1301  1 !
;   804          1302  1 ! Returned Value:
;   805          1303  1 !       none
;   806          1304  1 !
;   807          1305  1 ! Side Effects:
;   808          1306  1 !
;   809          1307  1 !--
;   810          1308  1
;   811          1309  1
;   812          1310  2 global routine anl$format_protection_mask(indent_level,message,protection): novalue = begin
;   813          1311  2
;   814          1312  2 own
;   815          1313  2       protection_table: vector[16,long] initial(
;   816          1314  2                                       uplit byte (%ascic 'RWED'),
;   817          1315  2                                       uplit byte (%ascic 'WED'),
;   818          1316  2                                       uplit byte (%ascic 'RED'),
;   819          1317  2                                       uplit byte (%ascic 'ED'),
;   820          1318  2                                       uplit byte (%ascic 'RWD'),
;   821          1319  2                                       uplit byte (%ascic 'WD'),
;   822          1320  2                                       uplit byte (%ascic 'RD'),
;   823          1321  2                                       uplit byte (%ascic 'D'),
;   824          1322  2                                       uplit byte (%ascic 'RWE'),
;   825          1323  2                                       uplit byte (%ascic 'WE'),
;   826          1324  2                                       uplit byte (%ascic 'RE'),
;   827          1325  2                                       uplit byte (%ascic 'E'),
;   828          1326  2                                       uplit byte (%ascic 'RW'),
;   829          1327  2                                       uplit byte (%ascic 'W'),
;   830          1328  2                                       uplit byte (%ascic 'R'),
;   831          1329  2                                       uplit byte (%ascic ''));
;   832          1330  2
;   833          1331  2
;   834          1332  2 ! Simply format the message using the above protection code table.
;   835          1333  2
;   836          1334  2 anl$format_line(0,.indent_level,.message,.protection_table[.protection<0,4,0>],
;   837          1335  2                                           .protection_table[.protection<4,4,0>],
;   838          1336  2                                           .protection_table[.protection<8,4,0>],
;   839          1337  2                                           .protection_table[.protection<12,4,0>]);
;   840          1338  2
;   841          1339  2 return;
;   842          1340  2
;   843          1341  1 end;
```

```
                                                                    .PSECT  $PLIT$,NOWRT,NOEXE,2

                                   44  45  57  52  04  00068 P.AAO:  .ASCII  <4>\RWED\                                  ;
                                       44  45  57  03  0006D P.AAP:  .ASCII  <3>\WED\                                   ;
                                       44  45  52  03  00071 P.AAQ:  .ASCII  <3>\RED\                                   ;
                                           44  45  02  00075 P.AAR:  .ASCII  <2>\ED\                                    ;
                                       44  57  52  03  00078 P.AAS:  .ASCII  <3>\RWD\                                   ;
                                           44  57  02  0007C P.AAT:  .ASCII  <2>\WD\                                    ;
                                           44  52  02  0007F P.AAU:  .ASCII  <2>\RD\                                    ;
                                               44  01  00082 P.AAV:  .ASCII  <1>\D\                                     ;
                                       45  57  52  03  00084 P.AAW:  .ASCII  <3>\RWE\                                   ;
                                           45  57  02  00088 P.AAX:  .ASCII  <2>\WE\                                    ;
                                           45  52  02  0008B P.AAY:  .ASCII  <2>\RE\                                    ;
                                               45  01  0008E P.AAZ:  .ASCII  <1>\E\                                     ;
                                           57  52  02  00090 P.ABA:  .ASCII  <2>\RW\                                    ;
                                               57  01  00093 P.ABB:  .ASCII  <1>\W\                                     ;
                                               52  01  00095 P.ABC:  .ASCII  <1>\R\                                     ;
                                                   00  00097 P.ABD:  .ASCII  <0>                                        ;

                                                                    .PSECT  $OWN$,NOEXE,2

                                                       00685         .BLKB   3
00000000' 00000000' 00000000' 00000000' 00000000' 00000000' 00688 PROTECTION_TABLE:
                                                                    .ADDRESS P.AAO, P.AAP, P.AAQ, P.AAR, P.AAS, -       ;
00000000' 00000000' 00000000' 00000000' 00000000' 00000000' 006A0            P.AAT, P.AAU, P.AAV, P.AAW, P.AAX, P.AAY, - ;
                    00000000' 00000000' 00000000' 00000000' 006B8            P.AAZ, P.ABA, P.ABB, P.ABC, P.ABD           ;


                                                                    .PSECT  $CODE$,NOWRT,2

                                            0004 00000               .ENTRY  ANL$FORMAT_PROTECTION_MASK, Save R2        ; 1310
                          52       0000'  CF  9E 00002               MOVAB   PROTECTION_TABLE, R2                       ;
       50      0D   AC    04                04  EF 00007               EXTZV   #4, #4, PROTECTION+1, R0                   ; 1337
                                          6240  DD 0000D               PUSHL   PROTECTION_TABLE[R0]                       ;
       50      0D   AC    04                00  EF 00010               EXTZV   #0, #4, PROTECTION+1, R0                   ; 1336
                                          6240  DD 00016               PUSHL   PROTECTION_TABLE[R0]                       ;
       50      0C   AC    04                04  EF 00019               EXTZV   #4, #4, PROTECTION, R0                     ; 1335
                                          6240  DD 0001F               PUSHL   PROTECTION_TABLE[R0]                       ;
       50      0C   AC    04                00  EF 00022               EXTZV   #0, #4, PROTECTION, R0                     ; 1334
                                          6240  DD 00028               PUSHL   PROTECTION_TABLE[R0]                       ;
                          7E         04   AC  7D 0002B               MOVQ    INDENT_LEVEL, -(SP)                        ;
                                            7E  D4 0002F               CLRL    -(SP)                                      ;
                    FCEE  CF                07  FB 00031               CALLS   #7, ANL$FORMAT_LINE                        ;
                                                04 00036               RET                                                ; 1341

; Routine Size:  55 bytes,    Routine Base:  $CODE$ + 04AE
```

```
;   845          1342  1 %sbttl 'ANL$FORMAT_FILE_ATTRIBUTES - Format File Attribute Area'
;   846          1343  1 !++
;   847          1344  1 ! Functional Description:
;   848          1345  1 !       This routine is called to format the user file attribute area, which
;   849          1346  1 !       is assumed to contain RMS file attributes.  We don't check the
;   850          1347  1 !       attributes.
;   851          1348  1 !
;   852          1349  1 ! Formal Parameters:
;   853          1350  1 !       none
;   854          1351  1 !
;   855          1352  1 ! Implicit Inputs:
;   856          1353  1 !       global data
;   857          1354  1 !
;   858          1355  1 ! Implicit Outputs:
;   859          1356  1 !       global data
;   860          1357  1 !
;   861          1358  1 ! Returned Value:
;   862          1359  1 !       none
;   863          1360  1 !
;   864          1361  1 ! Side Effects:
;   865          1362  1 !
;   866          1363  1 !--
;   867          1364  1
;   868          1365  1
;   869          1366  2 global routine anl$format_file_attributes: novalue = begin
;   870          1367  2
;   871          1368  2
;   872          1369  2 ! We start with a nice little header.
;   873          1370  2
;   874          1371  2 anl$format_line(3,0,anlrms$_fileattr);
;   875          1372  2 anl$format_skip(0);
;   876          1373  2
;   877          1374  2 !  The first data printed is the file organization.
;   878          1375  2
;   879          1376  3 anl$format_line(0,1,anlrms$_fileorg,(selectoneu .anl$gl_fat[fat$v_fileorg] of set
;   880          1377  3                                       [fat$c_sequential]:   uplit byte (%ascic 'sequential');
;   881          1378  3                                       [fat$c_relative]:     uplit byte (%ascic 'relative');
;   882          1379  3                                       [fat$c_indexed]:      uplit byte (%ascic 'indexed');
;   883          1380  2                                       tes));
;   884          1381  2
;   885          1382  2 ! Now we include the record format and attributes.
;   886          1383  2
;   887          1384  2 anl$format_line(0,1,anlrms$_recfmt,
;   888          1385  3                  (selectoneu .anl$gl_fat[fat$v_rtype] of set
;   889          1386  3                   [fat$c_undefined]:     uplit byte (%ascic 'undefined');
;   890          1387  3                   [fat$c_fixed]:         uplit byte (%ascic 'fixed');
;   891          1388  3                   [fat$c_variable]:      uplit byte (%ascic 'variable');
;   892          1389  3                   [fat$c_vfc]:           uplit byte (%ascic 'variable-with-fixed-control');
;   893          1390  3                   [fat$c_stream]:        uplit byte (%ascic 'stream');
;   894          1391  3                   [fat$c_streamlf]:      uplit byte (%ascic 'stream-LF');
;   895          1392  3                   [fat$c_streamcr]:      uplit byte (%ascic 'stream-CR');
;   896          1393  2                   tes));
;   897          1394  2
;   898          1395  3 anl$format_line(0,1,anlrms$_recattr,(if .anl$gl_fat[fat$v_nospan] then uplit byte (%ascic 'no-span')
;   899          1396  2                                      else
;   900          1397  3                                          (if .anl$gl_fat[fat$v_impliedcc] then uplit byte (%ascic 'carriage-return')
;   901          1398  3                                      else if .anl$gl_fat[fat$v_fortrancc] then uplit byte (%ascic 'fortran')
```

```
;  902          1399  3                                 else if .anl$gl_fat[fat$v_printcc]   then uplit byte (%ascic 'print')
;  903          1400  2                                 else                                      uplit byte (%ascic '')));
;  904          1401  2
;  905          1402  2 ! Now the maximum record size and the longest record size.
;  906          1403  2
;  907          1404  2 anl$format_line(0,1,anlrms$_maxrecsize,.anl$gl_fat[fat$w_maxrec]);
;  908          1405  2 if .anl$gl_fat[fat$v_fileorg] eqlu fat$c_sequential or .anl$gl_fat[fat$v_rtype] eqlu fat$c_fixed then
;  909          1406  2         anl$format_line(0,1,anlrms$_longrec,.anl$gl_fat[fat$w_rsize]);
;  910          1407  2
;  911          1408  2 ! Now the header size for variable with fixed control.
;  912          1409  2
;  913          1410  2 if .anl$gl_fat[fat$v_rtype] eqlu fat$c_vfc then
;  914          1411  2         anl$format_line(0,1,anlrms$_ct[size,.anl$gl_fat[fat$b_vfcsize]);
;  915          1412  2
;  916          1413  2 ! Now the number of blocks allocated, extend quantity, and the end-of-file
;  917          1414  2 ! information.
;  918          1415  2
;  919          1416  2 anl$format_line(0,1,anlrms$_alloc,.anl$gl_fat[fat$l_hiblk],.anl$gl_fat[fat$w_defext]);
;  920          1417  2 if .anl$gl_fat[fat$v_fileorg] eqlu fat$c_sequential then
;  921          1418  2         anl$format_line(0,1,anlrms$_eof,.anl$gl_fat[fat$l_efblk],.anl$gl_fat[fat$w_ffbyte]);
;  922          1419  2
;  923          1420  2 ! Now the bucket size, unless it's a sequential file.
;  924          1421  2
;  925          1422  2 if .anl$gl_fat[fat$v_fileorg] eqlu fat$c_relative or .anl$gl_fat[fat$v_fileorg] eqlu fat$c_indexed then
;  926          1423  2         anl$format_line(0,1,anlrms$_bucketsize,.anl$gl_fat[fat$b_bktsize]);
;  927          1424  2
;  928          1425  2 ! Finally, display the global buffer count.
;  929          1426  2
;  930          1427  2 anl$format_line(0,1,anlrms$_globalbufs,.anl$gl_fat[fat$w_gbc]);
;  931          1428  2
;  932          1429  2 return;
;  933          1430  2
;  934          1431  1 end;


                                                                                .PSECT  $PLIT$,NOWRT,NOEXE,2

                    6C  61  69  74  6E  65  75  71  65  73  0A  00098 P.ABE:   .ASCII  <10>\sequential\
                        65  76  69  74  61  6C  65  72  08  000A3 P.ABF:   .ASCII  <8>\relative\
                            64  65  78  65  64  6E  69  07  000AC P.ABG:   .ASCII  <7>\indexed\
                        64  65  6E  69  66  65  64  6E  75  09  000B4 P.ABH:   .ASCII  <9>\undefined\
                                    64  65  78  69  66  05  000BE P.ABI:   .ASCII  <5>\fixed\
                            65  6C  62  61  69  72  61  76  08  000C4 P.ABJ:   .ASCII  <8>\variable\
2D  68  74  69  77  2D  65  6C  62  61  69  72  61  76  1B  000CD P.ABK:   .ASCII  <27>\variable-with-fixed-control\
        6C  6F  72  74  6E  6F  63  2D  64  65  78  69  66  000DC
                                6D  61  65  72  74  73  06  000E9 P.ABL:   .ASCII  <6>\stream\
                        46  4C  2D  6D  61  65  72  74  73  09  000F0 P.ABM:   .ASCII  <9>\stream-LF\
                        52  43  2D  6D  61  65  72  74  73  09  000FA P.ABN:   .ASCII  <9>\stream-CR\
                                6E  61  70  73  2D  6F  6E  07  00104 P.ABO:   .ASCII  <7>\no-span\
                                                            00  0010C P.ABP:   .ASCII  <0>
72  75  74  65  72  2D  65  67  61  69  72  72  61  63  0F  0010D P.ABQ:   .ASCII  <15>\carriage-return\
                                                            6E  0011C
                                6E  61  72  74  72  6F  66  07  0011D P.ABR:   .ASCII  <7>\fortran\
                                        74  6E  69  72  70  05  00125 P.ABS:   .ASCII  <5>\print\
                                                            00  0012B P.ABT:   .ASCII  <0>
```

```
                                                                 .PSECT  $CODE$,NOWRT,2

                                          001C 00000             .ENTRY  ANL$FORMAT_FILE_ATTRIBUTES, Save R2,R3,R4    ; 1366
                       54      0000G  CF    9E 00002             MOVAB   ANL$GL_FAT, R4                               ;
                       53      FCE2   CF    9E 00007             MOVAB   ANL$FORMAT_LINE, R3                          ;
                       52      0000'  CF    9E 0000C             MOVAB   P.ABE, R2                                    ;
                          00000000G   8F    DD 00011             PUSHL   #ANLRMS$_FILEATTR                            ; 1371
                       7E             03    7D 00017             MOVQ    #3, -(SP)                                    ;
                       63             03    FB 0001A             CALLS   #3, ANL$FORMAT_LINE                          ;
                                      7E    D4 0001D             CLRL    -(SP)                                        ; 1372
                  00F6 C3             01    FB 0001F             CALLS   #1, ANL$FORMAT_SKIP                          ;
  51      00  B4       04             04    EF 00024             EXTZV   #4, #4, @ANL$GL_FAT, R1                      ; 1376
                                      05    12 0002A             BNEQ    1$                                           ; 1377
                       50             62    9E 0002C             MOVAB   P.ABE, R0                                    ;
                                      19    11 0002F             BRB     4$                                           ;
                       01             51    D1 00031 1$:         CMPL    R1, #1                                       ; 1378
                                      06    12 00034             BNEQ    2$                                           ;
                       50      08     A2    9E 00036             MOVAB   P.ABF, R0                                    ;
                                      0E    11 0003A             BRB     4$                                           ;
                       02             51    D1 0003C 2$:         CMPL    R1, #2                                       ; 1379
                                      05    13 0003F             BEQL    3$                                           ;
                       7E             01    CE 00041             MNEGL   #1, -(SP)                                    ;
                                      06    11 00044             BRB     5$                                           ;
                       50      14     A2    9E 00046 3$:         MOVAB   P.ABG, R0                                    ;
                                      50    DD 0004A 4$:         PUSHL   R0                                           ;
                          00000000G   8F    DD 0004C 5$:         PUSHL   #ANLRMS$_FILEORG                             ; 1376
                                      01    DD 00052             PUSHL   #1                                           ;
                                      7E    D4 00054             CLRL    -(SP)                                        ;
                       63             04    FB 00056             CALLS   #4, ANL$FORMAT_LINE                          ;
  50      00  B4       04             00    EF 00059             EXTZV   #0, #4, @ANL$GL_FAT, R0                      ; 1385
                                      06    12 0005F             BNEQ    6$                                           ; 1386
                       51      1C     A2    9E 00061             MOVAB   P.ABH, R1                                    ;
                                      35    11 00065             BRB     11$                                          ;
                       01             50    D1 00067 6$:         CMPL    R0, #1                                       ; 1387
                                      06    12 0006A             BNEQ    7$                                           ;
                       51      26     A2    9E 0006C             MOVAB   P.ABI, R1                                    ;
                                      2A    11 00070             BRB     11$                                          ;
                       02             50    D1 00072 7$:         CMPL    R0, #2                                       ; 1388
                                      06    12 00075             BNEQ    8$                                           ;
                       51      2C     A2    9E 00077             MOVAB   P.ABJ, R1                                    ;
                                      1F    11 0007B             BRB     11$                                          ;
                       03             50    D1 0007D 8$:         CMPL    R0, #3                                       ; 1389
                                      06    12 00080             BNEQ    9$                                           ;
                       51      35     A2    9E 00082             MOVAB   P.ABK, R1                                    ;
                                      14    11 00086             BRB     11$                                          ;
                       04             50    D1 00088 9$:         CMPL    R0, #4                                       ; 1390
                                      06    12 0008B             BNEQ    10$                                          ;
                       51      51     A2    9E 0008D             MOVAB   P.ABL, R1                                    ;
                                      09    11 00091             BRB     11$                                          ;
                       05             50    D1 00093 10$:        CMPL    R0, #5                                       ; 1391
                                      08    12 00096             BNEQ    12$                                          ;
                       51      58     A2    9E 00098             MOVAB   P.ABM, R1                                    ;
                                      51    DD 0009C 11$:        PUSHL   R1                                           ;
                                      10    11 0009E             BRB     14$                                          ;
                       06             50    D1 000A0 12$:        CMPL    R0, #6                                       ; 1392
```

```
                               05  13 000A3        BEQL    13$                              ;
                    7E         01  CE 000A5        MNEGL   #1, -(SP)                        ;
                               06  11 000A8        BRB     14$                              ;
                    50    62   A2  9E 000AA 13$:   MOVAB   P.ABN, R0                        ;
                               50  DD 000AE        PUSHL   R0                               ;
               00000000G       8F  DD 000B0 14$:   PUSHL   #ANLRMS$_RECFMT                  ; 1384
                               01  DD 000B6        PUSHL   #1                               ;
                               7E  D4 000B8        CLRL    -(SP)                            ;
                    63         04  FB 000BA        CALLS   #4, ANL$FORMAT_LINE              ;
                    50         64  D0 000BD        MOVL    ANL$GL_FAT, R0                   ; 1397
          06    01  A0         01  E1 000C0        BBC     #1, 1(R0), 15$                   ;
                    51    75   A2  9E 000C5        MOVAB   P.ABQ, R1                        ;
                               1C  11 000C9        BRB     18$                              ;
                    07    01   A0  E9 000CB 15$:   BLBC    1(R0), 16$                       ; 1398
                    51  0085   C2  9E 000CF        MOVAB   P.ABR, R1                        ;
                               11  11 000D4        BRB     18$                              ;
          07    01  A0         02  E1 000D6 16$:   BBC     #2, 1(R0), 17$                   ; 1399
                    51  008D   C2  9E 000DB        MOVAB   P.ABS, R1                        ;
                               05  11 000E0        BRB     18$                              ;
                    51  0093   C2  9E 000E2 17$:   MOVAB   P.ABT, R1                        ; 1400
                               51  DD 000E7 18$:   PUSHL   R1                               ; 1398
          06    01  A0         03  E1 000E9        BBC     #3, 1(R0), 19$                   ; 1395
                    50    6C   A2  9E 000EE        MOVAB   P.ABO, R0                        ;
                               04  11 000F2        BRB     20$                              ;
                    50    74   A2  9E 000F4 19$:   MOVAB   P.ABP, R0                        ; 1396
                               50  DD 000F8 20$:   PUSHL   R0                               ;
               00000000G       8F  DD 000FA        PUSHL   #ANLRMS$_RECATTR                 ; 1395
                               01  DD 00100        PUSHL   #1                               ;
                               7E  D4 00102        CLRL    -(SP)                            ;
                    63         05  FB 00104        CALLS   #5, ANL$FORMAT_LINE              ;
                    50         64  D0 00107        MOVL    ANL$GL_FAT, R0                   ; 1404
                    7E    10   A0  3C 0010A        MOVZWL  16(R0), -(SP)                    ;
               00000000G       8F  DD 0010E        PUSHL   #ANLRMS$_MAXRECSIZE              ;
                               01  DD 00114        PUSHL   #1                               ;
                               7E  D4 00116        CLRL    -(SP)                            ;
                    63         04  FB 00118        CALLS   #4, ANL$FORMAT_LINE              ;
                    50         64  D0 0011B        MOVL    ANL$GL_FAT, R0                   ; 1405
                F0  8F         60  93 0011E        BITB    (R0), #240                       ;
                               07  13 00122        BEQL    21$                              ;
     01         60  04         00  ED 00124        CMPZV   #0, #4, (R0), #1                 ;
                               11  12 00129        BNEQ    22$                              ;
                    7E    02   A0  3C 0012B 21$:   MOVZWL  2(R0), -(SP)                     ; 1406
               00000000G       8F  DD 0012F        PUSHL   #ANLRMS$_LONGREC                 ;
                               01  DD 00135        PUSHL   #1                               ;
                               7E  D4 00137        CLRL    -(SP)                            ;
                    63         04  FB 00139        CALLS   #4, ANL$FORMAT_LINE              ;
                    50         64  D0 0013C 22$:   MOVL    ANL$GL_FAT, R0                   ; 1410
     03         60  04         00  ED 0013F        CMPZV   #0, #4, (R0), #3                 ;
                               11  12 00144        BNEQ    23$                              ;
                    7E    0F   A0  9A 00146        MOVZBL  15(R0), -(SP)                    ; 1411
               00000000G       8F  DD 0014A        PUSHL   #ANLRMS$_CTLSIZE                 ;
                               01  DD 00150        PUSHL   #1                               ;
                               7E  D4 00152        CLRL    -(SP)                            ;
                    63         04  FB 00154        CALLS   #4, ANL$FORMAT_LINE              ;
                    50         64  D0 00157 23$:   MOVL    ANL$GL_FAT, R0                   ; 1416
                    7E    12   A0  3C 0015A        MOVZWL  18(R0), -(SP)                    ;
                          04   A0  DD 0015E        PUSHL   4(R0)                            ;
```

```
                                 00000000G  8F  DD 00161          PUSHL   #ANLRMS$_ALLOC                      ;
                                            01  DD 00167          PUSHL   #1                                  ;
                                            7E  D4 00169          CLRL    -(SP)                               ;
                            63              05  FB 0016B          CALLS   #5, ANL$FORMAT_LINE                 ;
                            50              64  D0 0016E          MOVL    ANL$GL_FAT, R0                      ; 1417
                       F0   8F              60  93 00171          BITB    (R0), #240                          ;
                                            14  12 00175          BNEQ    24$                                 ;
                            7E        0C    A0  3C 00177          MOVZWL  12(R0), -(SP)                       ; 1418
                                      08    A0  DD 0017B          PUSHL   8(R0)                               ;
                                 00000000G  8F  DD 0017E          PUSHL   #ANLRMS$_EOF                        ;
                                            01  DD 00184          PUSHL   #1                                  ;
                                            7E  D4 00186          CLRL    -(SP)                               ;
                            63              05  FB 00188          CALLS   #5, ANL$FORMAT_LINE                 ;
                            50              64  D0 0018B 24$:     MOVL    ANL$GL_FAT, R0                      ; 1422
         01          60     04              04  ED 0018E          CMPZV   #4, #4, (R0), #1                    ;
                                            07  13 00193          BEQL    25$                                 ;
         02          60     04              04  ED 00195          CMPZV   #4, #4, (R0), #2                    ;
                                            11  12 0019A          BNEQ    26$                                 ;
                            7E        0E    A0  9A 0019C 25$:     MOVZBL  14(R0), -(SP)                       ; 1423
                                 00000000G  8F  DD 001A0          PUSHL   #ANLRMS$_BUCKETSIZE                 ;
                                            01  DD 001A6          PUSHL   #1                                  ;
                                            7E  D4 001A8          CLRL    -(SP)                               ;
                            63              04  FB 001AA          CALLS   #4, ANL$FORMAT_LINE                 ;
                            50              64  D0 001AD 26$:     MOVL    ANL$GL_FAT, R0                      ; 1427
                            7E        14    A0  3C 001B0          MOVZWL  20(R0), -(SP)                       ;
                                 00000000G  8F  DD 001B4          PUSHL   #ANLRMS$_GLOBALBUFS                 ;
                                            01  DD 001BA          PUSHL   #1                                  ;
                                            7E  D4 001BC          CLRL    -(SP)                               ;
                            63              04  FB 001BE          CALLS   #4, ANL$FORMAT_LINE                 ;
                                                04 001C1          RET                                         ; 1431

; Routine Size:  450 bytes,    Routine Base:  $CODE$ + 04E5


;  935          1432  1
;  936          1433  0 end eludom



                                                                  .EXTRN  LIB$SIGNAL

;                                  PSECT SUMMARY
;
;       Name                    Bytes                      Attributes
;
;  $OWN$                         1736  NOVEC,  WRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  $PLIT$                         300  NOVEC,NOWRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  $CODE$                        1703  NOVEC,NOWRT,  RD ,  EXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)



;                             Library Statistics
;
;                                    -------- Symbols --------      Pages       Processing
;       File                         Total    Loaded   Percent      Mapped      Time
```

```
;
;  _$255$DUA28:[SYSLIB]LIB.L32;1              18619        84         0       1000          00:01.8




;                                        COMMAND QUALIFIERS

;       BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/LIS=LIS$:RMSREPORT/OBJ=OBJ$:RMSREPORT MSRC$:RMSREPORT/UPDATE=(ENH$:RMSREPORT)

; Size:          1703 code + 2036 data bytes
; Run Time:        00:34.3
; Elapsed Time:    01:49.7
; Lines/CPU Min:    2507
; Lexemes/CPU-Min: 21226
; Memory Used:  264 pages
; Compilation Complete
```

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

BACKDEF
SDL

COMMON
REQ

BACKUP

STABACCOP
MAP

BACKUP
MAP

RMSREPORT
LIS

RMSSTATS
LIS

STABACKUP
MAP